|  |  |  |
| --- | --- | --- |
| wordml://75.png |  | wordml://76.png |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | |  | | --- | | Imię i nazwisko studenta: Jakub Warzyński | | | |  | | --- | | Nr albumu: 165550 | | | |  | | --- | | Studia drugiego stopnia | | | |  | | --- | | Forma studiów: stacjonarne | | | |  | | --- | | Kierunek studiów: Informatyka | | | specjalizacja: Aplikacje Rozproszone i Systemy Internetowe | |
|  |
|  |
|  |
| |  | | --- | | **PROJEKT DYPLOMOWY MAGISTERSKI** | |
| |  | | --- | | Tytuł projektu w języku polskim: Baza danych mikroflory jelitowej w grupie pacjentów z chorobami wątroby oraz powracających z krajów tropikalnych. |  |  | | --- | | Tytuł projektu w języku angielskim: Intestinal microflora database for patient group with liver diseases and returning from tropical countries. |  |  | | --- | |  | |
| |  |  |  |  | | --- | --- | --- | --- | |  | |  | | --- | | Potwierdzenie przyjęcia projektu | | | |  | |  | | --- | | Opiekun projektu |   *podpis* | |  | | --- | | Kierownik Katedry/Zakładu |   *podpis* | |  | dr inż. Marcin Kulawiak | dr hab. inż. | |
| |  | | --- | | Data oddania projektu do dziekanatu: | |

|  |  |  |  |
| --- | --- | --- | --- |
| wordml://75.png | |  | wordml://76.png |
|  | |  | | --- | | **OŚWIADCZENIE dotyczące projektu dyplomowego zatytułowanego:**  **Baza danych mikroflory jelitowej w grupie pacjentów z chorobami wątroby oraz powracających z krajów tropikalnych.** | | | |
|  | |  |  | | --- | --- | | |  | | --- | | Imię i nazwisko studenta: Jakub Warzyński  Data i miejsce urodzenia: 25.11.1997, Kołobrzeg  Nr albumu: 165550 | | | |  | | --- | | Wydział: Wydział Elektroniki, Telekomunikacji i Informatyki  Kierunek: informatyka | | | |  | | --- | | Poziom kształcenia: drugi  Forma studiów: stacjonarne | | | | |
|  | |  | | --- | | Świadomy(a) odpowiedzialności karnej z tytułu naruszenia przepisów ustawy z dnia 4 lutego 1994 r. o prawie autorskim i prawach pokrewnych (Dz. U. 2018 poz. 1191 z późn. zm.) i konsekwencji dyscyplinarnych określonych w ustawie z dnia 20 lipca 2018 r. Prawo o szkolnictwie wyższym i nauce (Dz. U. 2018 poz. 1668 z późn. zm.),1 a także odpowiedzialności cywilnoprawnej oświadczam, że przedkładany projekt dyplomowy został opracowany przeze mnie samodzielnie.  Niniejszy projekt dyplomowy nie był wcześniej podstawą żadnej innej urzędowej procedury związanej z nadaniem tytułu zawodowego.  Wszystkie informacje umieszczone w ww. projekcie dyplomowym, uzyskane ze źródeł pisanych i elektronicznych, zostały udokumentowane w wykazie literatury odpowiednimi odnośnikami zgodnie z art. 34 ustawy o prawie autorskim i prawach pokrewnych.  Potwierdzam zgodność niniejszej wersji projektu dyplomowego z załączoną wersją elektroniczną. | | | |
|  | |  |  | | --- | --- | | Gdańsk, dnia .................................. | .....................................................  *podpis studenta* | | | |

# STRESZCZENIE

**Słowa kluczowe:**

**Dziedzina nauki i techniki, zgodnie z wymogami OECD:** 1.2 Nauki o komputerach i informatyka

# ABSTRACT

**Keywords:**

**Field of Science and Technology, as required by OECD**: 1.2 Computer and information sciences

**SPIS TREŚCI**

[STRESZCZENIE 3](#_Toc80463201)

[ABSTRACT 4](#_Toc80463202)

[WYKAZ WAŻNIEJSZYCH OZNACZEŃ I SKRÓTÓW 7](#_Toc80463203)

[1. Wstęp i cel pracy 7](#_Toc80463204)

[2. Przegląd podobnych rozwiązań 8](#_Toc80463205)

[2.1 Centers for Disease Control and Prevention 8](#_Toc80463206)

[2.2 COVID-19 Dashboard by the Center for Systems Science and Engineering (CSSE) at Johns Hopkins University (JHU) 10](#_Toc80463207)

[2.3 World Health Organization 11](#_Toc80463208)

[2.4 Podsumowanie 14](#_Toc80463209)

[3. Specyfikacja wymagań systemowych 14](#_Toc80463210)

[3.1. Udziałowcy 14](#_Toc80463211)

[3.2. Cele systemu 15](#_Toc80463212)

[3.2.1. Cele biznesowe 15](#_Toc80463213)

[3.2.2. Cele funkcjonalne 15](#_Toc80463220)

[3.3. Użytkownicy 15](#_Toc80463221)

[3.4. Komponenty systemu 16](#_Toc80463222)

[3.4.1. Komponenty sprzętowe 16](#_Toc80463223)

[3.4.2. Komponenty programowe 16](#_Toc80463232)

[3.5. Wymagania funkcjonalne 17](#_Toc80463233)

[3.6. Wymaganie jakościowe 17](#_Toc80463234)

[3.7. Wymagania dodatkowe 18](#_Toc80463235)

[4. Projekt Systemu 18](#_Toc80463236)

[4.1 Architektura systemu 18](#_Toc80463237)

[4.2 Projekt warstwy danych 20](#_Toc80463243)

[4.3 Projekt warstwy logiki biznesowej 23](#_Toc80463244)

[4.3.1. Klasy ORM 26](#_Toc80463245)

[4.3.2. Klasy widoków 26](#_Toc80463246)

[4.3.3. Obsługa badań i wywiadów medycznych 28](#_Toc80463247)

[4.4 Projekt interfejsu użytkownika 30](#_Toc80463248)

[5. Testy systemu 36](#_Toc80463249)

[5.1 Testy wydajnościowe 37](#_Toc80463250)

[5.2 Testy akceptacyjne 40](#_Toc80463251)

[6. ??? 42](#_Toc80463252)

[7. Podsumowanie 42](#_Toc80463253)

[WYKAZ LITERATURY 42](#_Toc80463254)

[WYKAZ RYSUNKÓW 43](#_Toc80463255)

[WYKAZ TABEL 43](#_Toc80463256)

# WYKAZ WAŻNIEJSZYCH OZNACZEŃ I SKRÓTÓW

# Wstęp i cel pracy

Wykorzystanie technologii informatycznych w medycynie, zarówno w lecznictwie jak i obszarach badawczych, jest w czasach dzisiejszych praktyką pospolitą przynoszącą dużo korzyści, zarówno dla pacjentów jak i dla środowiska lekarskiego. Jednymi z takich zastosowań są systemy geoinformatyczne, pozwalające na analizę danych dotyczących występowania ognisk zakażeń w różnych miejscach na świecie, w celu efektywniejszego dostarczania pomocy chorym, prowadzeniu prewencji poprzez ograniczanie przepływu ludzi przez takie miejsca, lub przynajmniej uczulaniu ich na istniejące zagrożenia, jak i pogłębienia wiedzy dotyczących danych patogenów, co pomoże w walce z nimi przy kolejnych ich wystąpieniach.

Celem pracy jest realizacja właśnie takiego systemu, przy współpracy z przedstawicielami Gdańskiego Uniwersytetu Medycznego oraz Gdańskiego Uniwersyteckiego Centrum Klinicznego, a konkretnie jego placówki – Uniwersyteckiego Centrum Medycyny Morskiej i Tropikalnej w Gdyni. Byli oni głównymi akcjonariuszami systemu i dostarczali większość jego wymagań systemowych i pozasystemowych, w formie list wymogów, oraz kwestionariuszy przedstawianych pacjentom oraz lekarzom przeprowadzających badania, które miały zostać poddane cyfryzacji. System pozwala na zarządzanie bazą danych patogenów różnego rodzaju oraz pacjentów, przeprowadzanie analiz statystycznych na tych danych oraz eksport danych do popularnych formatów. Głównym wymogiem funkcjonalnym systemu jest możliwość przedstawienia wybranych danych dotyczących zakażeń występujących u konkretnej grupy pacjentów na mapie świata, co jest pomocnym narzędziem przy przeprowadzaniu analiz geoprzestrzennych. Dodatkowo system został zrealizowany w postaci przeglądarkowej aplikacji sieciowej z interfejsem graficznym oraz systemem kont użytkowników.

Konieczność realizacji takiego systemu uwidacznia się w momencie przeglądu podobnych rozwiązań dostępnych na rynku. Znaleźć można rozwiązania zapewniające gotowe raporty i opracowania wraz z reprezentacją danych w postaci geograficznej, jednak nie posiadają one możliwości wprowadzenia własnych danych lub przeprowadzenia analizy na podstawie specyficznych kryteriów. O ile umożliwia to przestudiowanie konkretnych sytuacji w skali globalnej, to nie zapewnia to możliwości wglądu w sytuację dotyczącą lokalnej grupy pacjentów, ani patogenów lub schorzeń nie będących głównym skupieniem światowych organizacji medycznych.

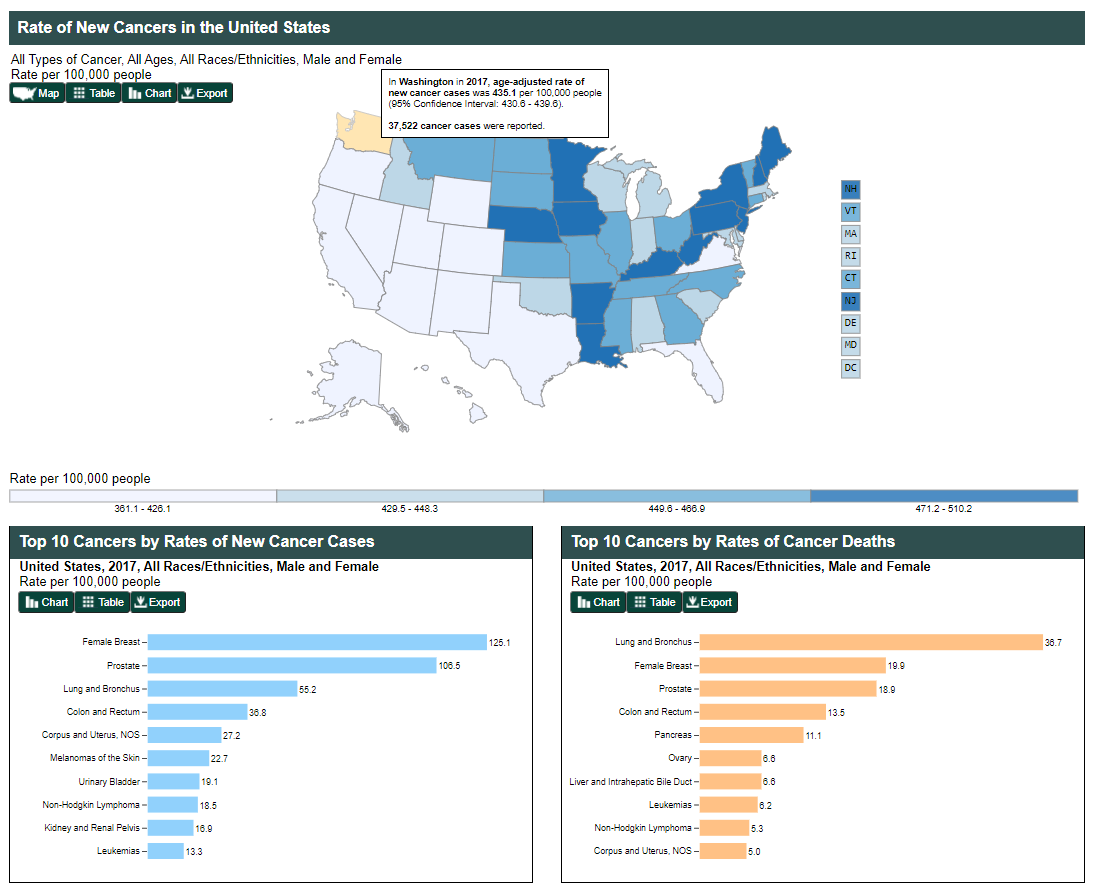
System jest przeznaczony do wdrożenia w Uniwersyteckim Centrum Medycyny Morskiej i Tropikalnej w Gdyni, przez co jego realizacja wiązała się z częstymi konsultacjami z przedstawicielami środowiska lekarskiego, co przekładało się na konieczność elastyczności pracy i rozwiązań, przy uwzględnieniu konieczności otrzymania zatwierdzenia każdej jego części przez pracowników szpitala oraz wprowadzania poprawek zgodnych ze zgłoszonymi uwagami. Dodatkowym utrudnieniem był tu fakt prowadzenia prac podczas pandemii. Potrzeba przekazania licencji na użytkowanie systemu w placówce medycznej wiązała się z ograniczeniem do technologii opartych na otwarte licencje.

Wszystkie założenia projektowe zostały zrealizowane i przeszły testy akceptacyjne u klienta. W kolejnych rozdziałach omówione zostały podobne rozwiązania geoinformatyczne w medycynie, specyfikacja wymagań systemowych, takich jak udziałowcy systemu, cele biznesowe, użytkownicy i komponenty systemu oraz wszystkie wymagania funkcjonalne i jakościowe. Omówiony został także projekt systemu, począwszy od jego architektury i dalej wszystkie jego warstwy – warstwę danych, warstwę logiki biznesowej i projekt interfejsu użytkownika. Przedstawiono proces testowania systemu pod kątem optymalizacyjnym wraz z wnioskami które z niego wynikły oraz opisano, jak przebiegały testy akcpetacyjne i jaki wpływ na nie miała panująca pandemia COVID-19.

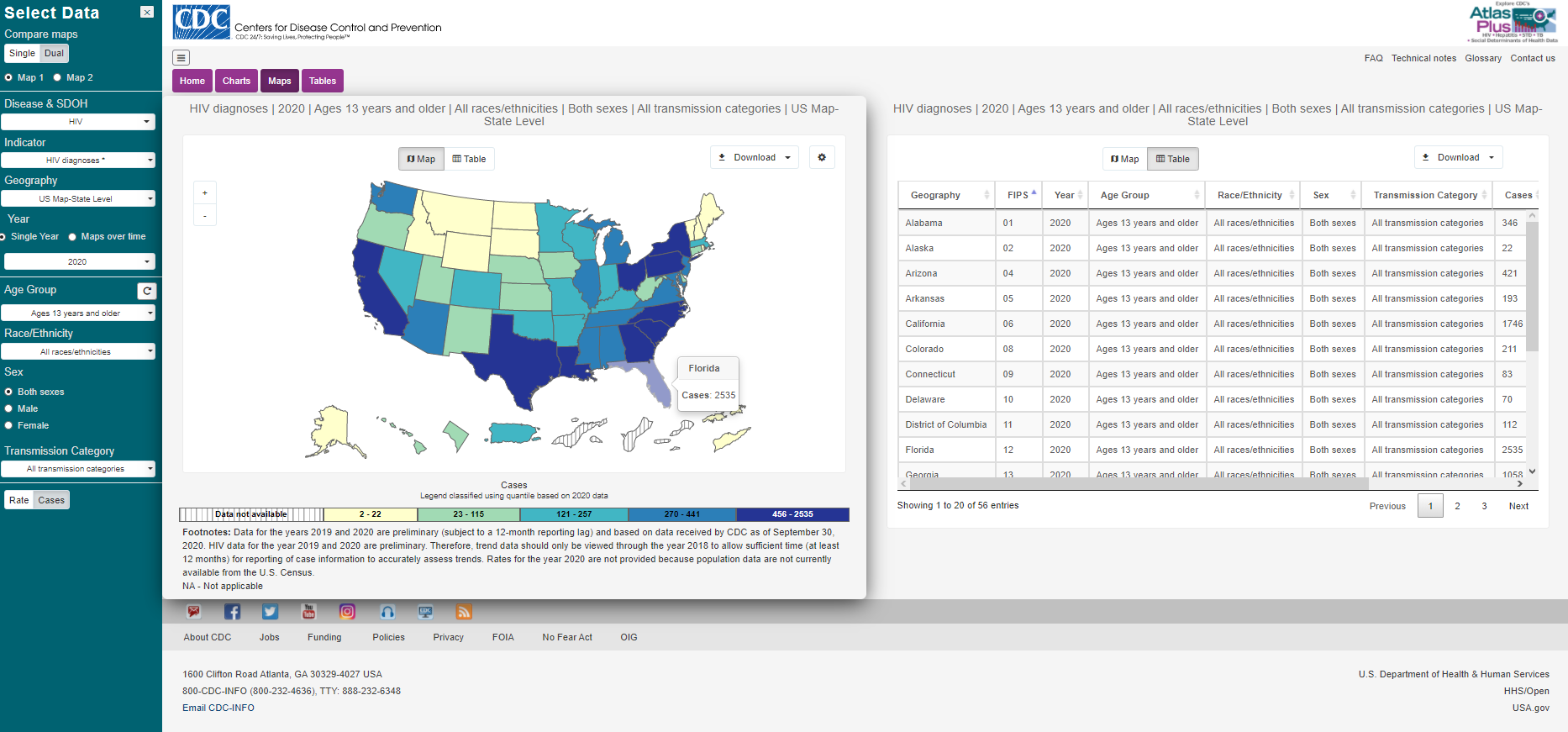
# Przegląd podobnych rozwiązań

W rozdziale wymieniono i omówiono kilka wybranych ogólnodostępnych rozwiązań geoinformatycznych w medycynie. Zwrócono uwagę na dostępne dane i analizy, sposób i szczegółowość ich przedstawienia, oraz na elementy wspólne wymienionych rozwiązań. Przegląd posłużył do wstępnego zaprojektowania architektury systemu, jak i identyfikacji mniej oczywistych problemów i przeszkód wiążących się z realizacją narzuconych wymagań systemowych.

## Centers for Disease Control and Prevention

Centers for Disease Control and Prevention (CDC) jest narodową agencją zdrowia Stanów Zjednoczonych. Jej głównym celem jest ochrona zdrowia i bezpieczeństwa poprzez kontrolę, prewencję i przeprowadzanie badań na temat chorób i uszczerbków na zdrowiu w USA i za granicą. Agencja prowadzi stronę internetową, na której zamieszczane są informacje dotyczące aktualnych epidemii na świecie, wiadomości o tematyce medycznej, raporty medyczne i artykuły naukowe o tematyce medycznej i wirusologicznej [1]. Po wybraniu przez użytkownika konkretnego schorzenia z menu znajdującego się w górnej części strony, prezentowane są informacje dotyczące tylko wybranej dolegliwości. Wśród zawartych raportów i narzędzi odnaleźć można liczne statystyki jak i ich graficzną reprezentację przedstawioną na mapach.  


Rys. 1 - CDC - Dane dotyczące nowotworu w USA przedstawione na mapie Kraju [2]

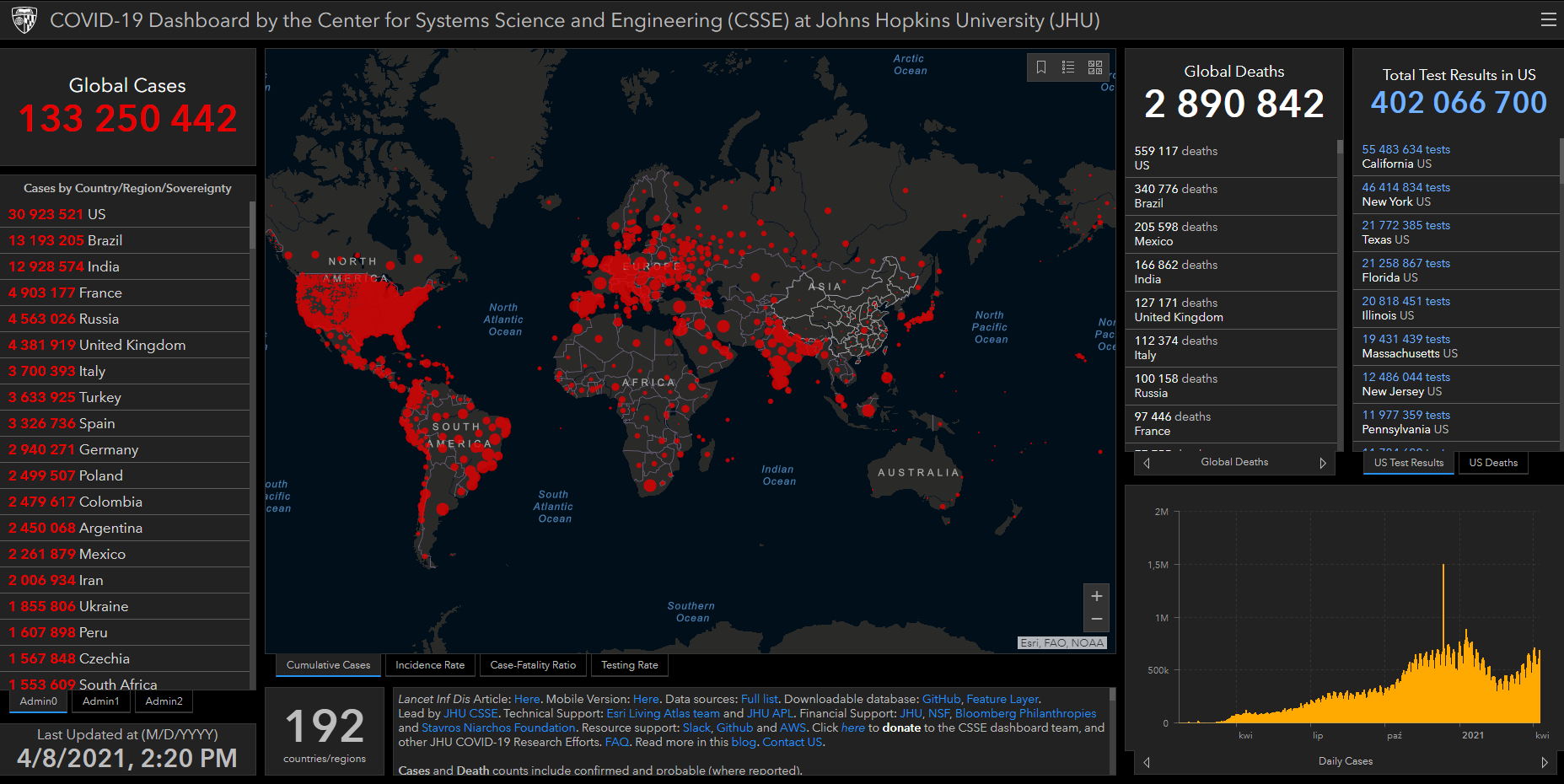
Rys. 2 - Dane dotyczące HIV w USA w roku 2020 przedstawione na mapie kraju i w tabelce [3]

Na rys.1 oraz rys.2 przedstawione zostały narzędzia znajdujące się na podstronach CDC, pozwalające na podgląd danych statystycznych dotyczących występowania kolejno nowotworów oraz HIV w Stanach Zjednoczonych, pogrupowanych według poszczególnych stanów.

Witryny dotyczące poszczególnych schorzeń są prowadzone przez oddzielne działy CDC. Skutkuje to niestety nieujednoliconymi projektami podstron i grupowaniami informacji, przez co nawigacja po stronie internetowej może być uciążliwa. Kolejnym minusem jest ograniczenie większości raportów i narzędzi do demografii USA, co sprawia że dla osoby zainteresowanej danymi dotyczących innych rejonów świata prezentowane materiały są niewystarczające.

## COVID-19 Dashboard by the Center for Systems Science and Engineering (CSSE) at Johns Hopkins University (JHU)

Duża część rozwiązań geoinformatycznych w medycynie jest projektowana z myślą o jednym konkretnym przypadku użycia. Pandemia wirusa COVID-19 szczególnie wzbudziła zainteresowanie zbieraniem danych w celach analizy i zainspirowała wiele osób do zaimplementowania systemów śledzących powstawanie nowych ognisk zakażeń, jak i przedstawiania statystyk względem poszczególnych rejonów świata. Jednym z takich rozwiązań jest COVID-19 Dashboard zaprojektowany przez członków Ceneter for System Science and Engineering (CSSE), wydziału na Johns Hopkins University w Stanach Zjednoczonych [4].



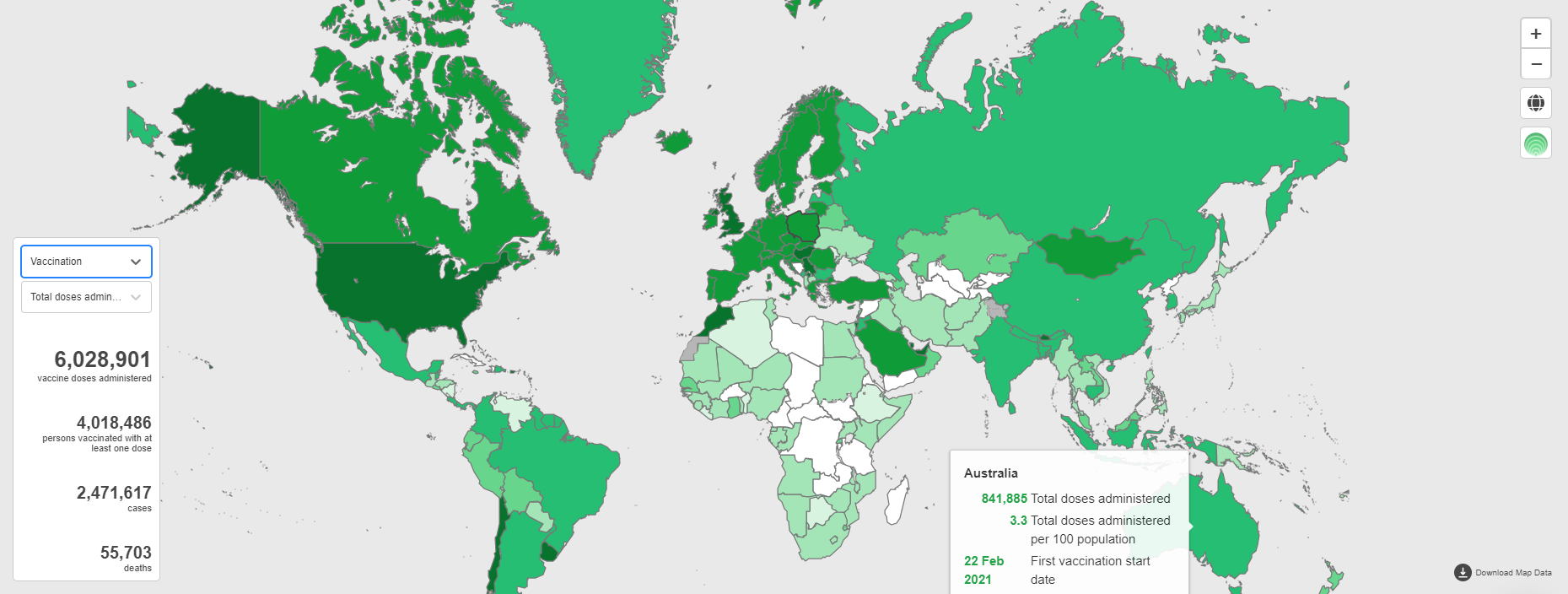
Rys. 3 - COVID-19 Dashboard [5].

Na rys. 3 przedstawiono dashboard, ogólnodostępny poprzez stronę internetową wydziału [5]. W centrum widoku znajduje się mapa świata, pod którą znajdują się opcje pozwalające na wyświetlenie na niej łącznych przypadków zakażeń, częstotliwości zakażeń, stosunku zakażeń do śmiertelności, oraz częstotliwości wykonywania testów obecności wirusa w organizmie (tylko dla Stanów Zjednoczonych). Użytkownik ma możliwość wyboru stylistyki mapy (dostępne opcje to między innymi widok nocny, satelitarny, lub z wyróżnieniem poszczególnych państw) oraz dowolnego przybliżania lub oddalania. Po lewej stronie znajdują się dane dotyczące globalnej ilości zgonów wskutek wirusa, razem z podziałem na poszczególne państwa (z możliwością wglądu w statystyki dotyczących poszczególnych stanów USA). W prawej części strony zamieszczone są informacje dotyczące światowych ilości zgonów i wyzdrowień, pogrupowane względem poszczególnych państw, liczba wykonanych testów na obecność COVID-19 w USA, pogrupowane względem stanów, oraz wykres przedstawiający dzienną liczbę zakażeń.

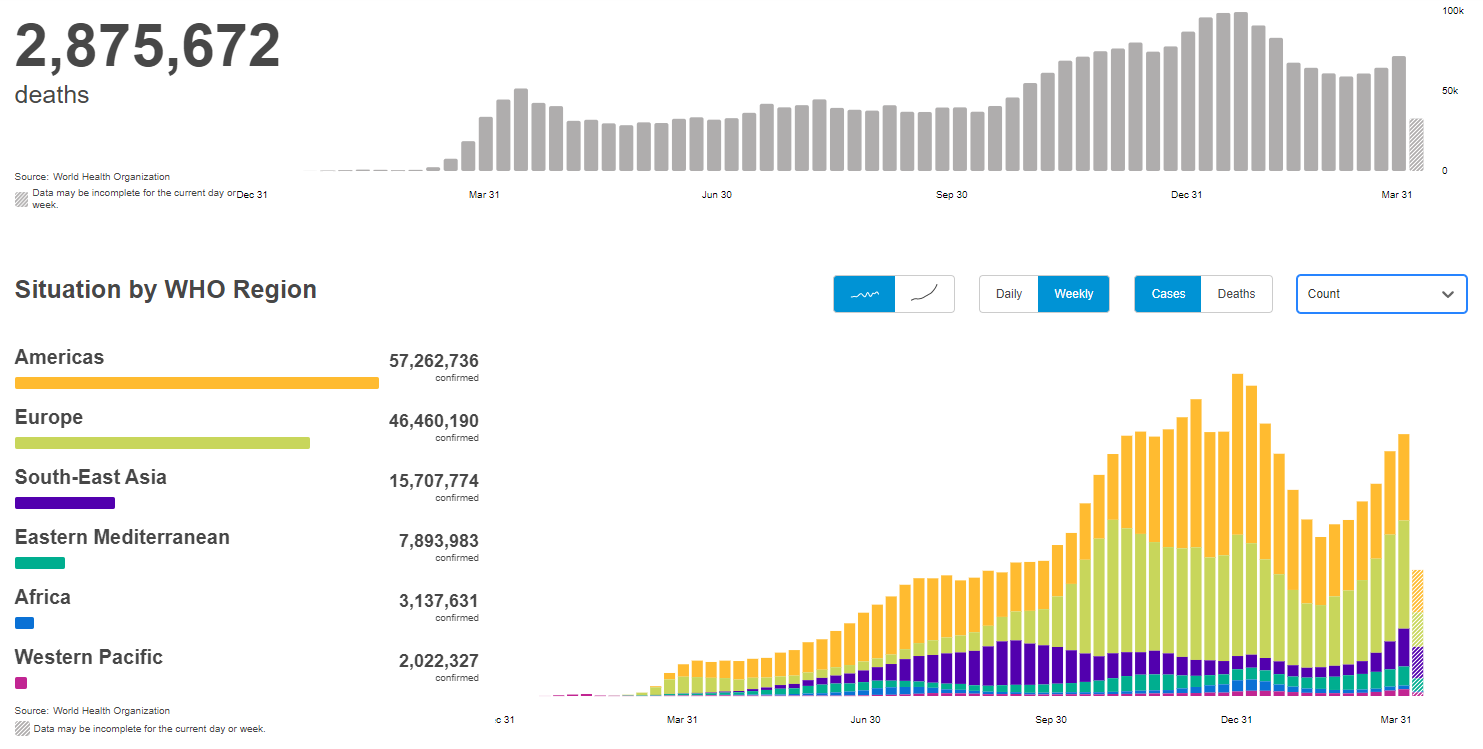
COVID-19 Dashboard jest bardzo rozbudowanym narzędziem do śledzenia stopnia rozprzestrzenienia się wirusa. Wszystkie dane przedstawione na stronie są udostępnione przez twórców na serwisie github [6].

## World Health Organization

Światowa Organizacja Zdrowia prowadzi stronę internetową [7] na której zamieszczone są liczne informacje, wiadomości oraz raporty z dziedziny światowej medycyny. WHO jest najbardziej znaną organizacją zajmującą się sprawami zdrowotnymi oraz prowadzą swoją działalność na całym świecie, przez co mają dostęp do danych z większości rejonów świata. Na stronie można znaleźć interaktywne narzędzia pozwalające na przejrzenie konkretnych danych na mapie świata, jak i rozległą liczbę raportów prezentujących liczne statystyki w postaci wykresów i map cieplnych.

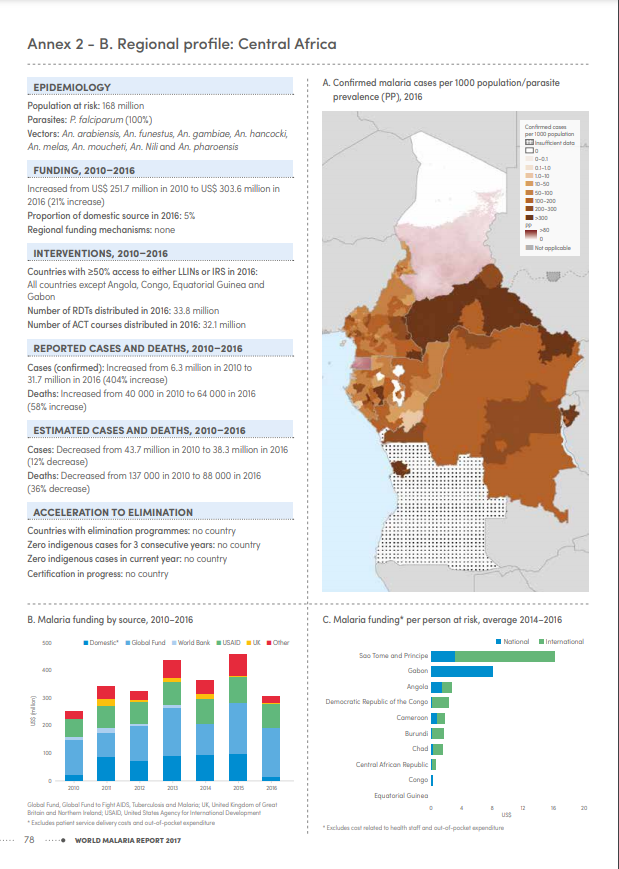


Rys. 4 - Coronavirus disease (COVID-19) Situation dashboard – mapa świata [8].



Rys. 5 - Coronavirus disease (COVID-19) situation dashboard – wykresy [8]

Na rys. 4 i rys. 5 przedstawiono jedno z takich narzędzi – panel dotyczący wirusa COVID-19. Użytkownik ma możliwość wyboru informacji przestawianych na mapie (ilość zarażeń, ilość zgonów, najpowszechniejszy rodzaj transmisji wirusa, oraz ilość szczepień) względem poszczególnych państw. Po lewej stronie mapy zamieszczone są informacje globalne dotyczące wybranej opcji. Dodatkowo pod mapą znajdują się wykresy, przedstawiające liczbę zakażeń i zgonów we wszystkich rejonach i państwach świata.



Rys. 6 - Raport WHO dotyczący stanu epidemiologicznego malarii w Afryce Centralnej [9].

Oprócz narzędzi interaktywnych, na stronie znajduje się także duża liczba raportów sporządzonych przez ekspertów. Zamieszczone w nich są szczegółowe informacje dotyczące omawianego zagadnienia, przedstawione na wykresach oraz na mapach. Na rys. 6 zamieszona została jedna ze stron z raportu dotyczącego sytuacji epidemiologicznej malarii na świecie przedstawiających sytuację w Afryce Centralnej. Widać na niej mapę cieplną rejonu, podzieloną na komórki odpowiadające poszczególnym regionom, wraz z wykresami przedstawiające ilość funduszy przeznaczonych na walkę z chorbą (ilość ogólną oraz ilość na osobę w strefie ryzyka).

## Podsumowanie

Przedstawione rozwiązania są bogatymi źródłami informacji z dziedziny epidemiologii. Pozwalają one na wgląd w szczegółowe dane, które przedstawione zostały w postaci licznych wykresów oraz informacji na mapach, co ułatwia ich przyswojenie, analizę i wyciągnięcie z nich wniosków. Niestety wszystkie wskazane implemenetacje wykazują pewne problemy. Największym z nich jest brak uniwersalności. Do dyspozycji są jedynie gotowe raporty, sporządzone przez osoby trzecie. W momencie w którym grupa osób chciałaby dokonać analizy danych zebranych własnoręcznie, wymienione wcześniej narzędzia nie nadawałyby się do przeprowadzenia takich działań. Dodatkowo, wszystkie wymienione organizacje zajmują się tylko wybranymi przypadkami schorzeń, zaś uwidacznia się w ich repozytoriach brak materiałów dotyczących bardziej szczególnych przypadków, jak na przykład przypadki zainfekowań pasożytem o konkretnym gatunku i subtypie. Kolejną wadą rozwiązań jest ich skupienie na jednym konkretnym przypadku, co wymusza zapoznanie się z dużą ilością narzędzi, znajdujących się pod różnymi witrynami oraz z różnymi interfejsami graficznymi, w przypadku w którym istnieje potrzeba przeprowadzenia analizy danych dotyczących wielu schorzeń jednocześnie. Dodatkowo nie wszystkie zawarte informacje odnoszą się do stanu globalnego, a jedynie skupiają się na poszczególnych rejonach świata.

Wynika z tego potrzeba zaprojektowania rozwiązania, które pozwalałoby zaagregować różne dane, niezależnie od ilości i rodzajów schorzeń, i przedstawić je na mapie świata w sposób jednolity, bez potrzeby używania wielu narzędzi i nauki nowych interfejsów i ich obsługi przez użytkowników.

# Specyfikacja wymagań systemowych

Wszystkie wymagania systemowe i biznesowe zostały zapewnione przez przedstawicieli Uniwersyteckiego Centrum Klinicznego w sposób bezpośredni, lub pośrednio poprzez dostarczone formularze z wywiadów medycznych z pacjentami, które należało poddać cyfryzacji. Dodatkowe wymagania dotyczące korzystania z systemu wynikają z potrzeby uwzględnienia faktu, że system będzie użytkowany przez osoby, które nie koniecznie muszą posiadać duże doświadczenie techniczne.

## Udziałowcy

|  |  |
| --- | --- |
| Udziałowiec: | Uniwerysteckie Centrum Kliniczne |
| Opis: | Zleceniodawca projektu |

|  |  |
| --- | --- |
| Udziałowiec: | Lekarze z Uniwerysteckiego Centrum Klinicznego |
| Opis: | Docelowi użytkownicy systemu |

|  |  |
| --- | --- |
| Udziałowiec: | Zespół projektowy |
| Opis: | Osoba pracująca nad rozwojem systemu |

## Cele systemu

### Cele biznesowe

|  |  |
| --- | --- |
| Cel: | Umożliwienie przeprowadzenia analizy statystycznej na danych dotyczących pacjentów |
| Opis: | Zapewnienie funkcjonalności pozwalających na przeprowadzenie analizy dotyczących pacjentów i wszelakich patogenów, ciał obcych i dolegliwości u nich wykrytych, oraz powiązanie tych informacji z lokalizacjami geograficznymi. |



### Cele funkcjonalne

|  |  |
| --- | --- |
| Cel: | Umożliwienie zarządzania bazą danych patogenów, ciał obcych i znanych dolegliwości |
| Opis: | Agregacja informacji dotyczących patogenów, ciał obcych i znanych dolegliwości wraz z możliwością manipulacji na nich oraz ich przeglądania z zastosowanymi filtrami. |

|  |  |
| --- | --- |
| Cel: | Umożliwienie zarządzania bazą danych pacjentów |
| Opis: | Agregacja informacji dotyczących pacjentów szpitala wraz z możliwością manipulacji na nich oraz ich przeglądania z zastosowanymi filtrami. |

|  |  |
| --- | --- |
| Cel: | Umożliwienie zarządzania bazą danych wywiadów medycznych z pacjentami |
| Opis: | Agregacja informacji dotyczących wywiadów medycznych z pacjentami wraz z możliwością manipulacji na nich oraz ich przeglądania z zastosowanymi filtrami. |

## Użytkownicy

|  |  |
| --- | --- |
| Użytkownik: | Stażysta |
| Opis: | Osoba przeprowadzająca staż lub szkolenia w szpitalu, posiadająca dostęp do przeglądania danych zawartych w systemie, ale nie do manipulacji na nich. |

|  |  |
| --- | --- |
| Użytkownik: | Lekarz |
| Opis: | Osoba posiadająca pełny dostęp do danych medycznych zawartych w systemie |

|  |  |
| --- | --- |
| Użytkownik: | Administrator |
| Opis: | Osoba posiadająca pełny dostęp do wszystkich danych zawartych w systemie oraz możliwość zarządzania jego użytkownikami. |

## Komponenty systemu

### Komponenty sprzętowe

|  |  |
| --- | --- |
| Komponent: | Serwer główny |
| Opis: | Główny serwer służący do utrzymania systemu. |

|  |  |
| --- | --- |
| Komponent: | Maszyny klienckie |
| Opis: | Komputery osobiste i mobilne użytkowników z możliwością uruchomienia przeglądarki internetowej. |



### Komponenty programowe

|  |  |
| --- | --- |
| Komponent: | Baza danych |
| Opis: | Baza danych zawierająca informacje dotyczące znanych patogenów, ciał obcych, schorzeń leków oraz badań morfologicznych, pacjentów szpitala wraz z ich wywiadami medycznymi, oraz użytkowników systemu |

|  |  |
| --- | --- |
| Komponent: | System autoryzacji |
| Opis: | Komponent odpowiedzialny za rejestrowanie, logowanie i autoryzowanie użytkowników systemu. |

|  |  |
| --- | --- |
| Komponent: | Interfejs użytkownika |
| Opis: | Interfejs dla użytkowników systemu, zrealizowany w postaci strony w przeglądarce internetowej. |

## Wymagania funkcjonalne

|  |  |
| --- | --- |
| Wymaganie: | Zarządzanie danymi w systemie |
| Opis: | Umożliwienie wykonywania operacji CRUD na danych w systemie. |

|  |  |
| --- | --- |
| Wymaganie: | Przedstawienie danych dotyczących wywiadów medycznych pacjentów na mapie świata |
| Opis: | Przedstawienie wybranych rekordów dotyczących wywiadów medycznych pacjentów na mapie świata w postaci kartogramu. |

|  |  |
| --- | --- |
| Wymaganie: | Filtrowanie danych zawartych w systemie |
| Opis: | Zaawansowane opcje filtrowania informacji zawartych w systemie pozwalające na szczegółową analizę w zależności od różnych czynników. |

|  |  |
| --- | --- |
| Wymaganie: | Sortowanie wyświetlanych danych |
| Opis: | Możliwość posortowania wyświetlanych danych na podstawie wybranych kolumn. |

|  |  |
| --- | --- |
| Wymaganie: | Pobranie danych zawartych w systemie |
| Opis: | Możliwość pobrania wyświetlanych danych przy zaaplikowanych filtrach i opcjach sortowania w formacie CSV. |

|  |  |
| --- | --- |
| Wymaganie: | Zarządzanie użytkownikami |
| Opis: | Administrator musi być w stanie zarządzać użytkownikami w systemie poprzez ich tworzenie i usuwanie. |

|  |  |
| --- | --- |
| Wymaganie: | Logowanie użytkowników |
| Opis: | Zalogowanie użytkownika i nadanie mu adekwatnych uprawnień. |

## Wymaganie jakościowe

|  |  |
| --- | --- |
| Wymaganie: | Przejrzysty graficzny interfejs użytkownika |
| Opis: | Jako że system ma być obsługiwany przez osoby nie posiadające dużego doświadczenia technicznego, konieczne jest dostarczenie interfejsu graficznego skonstruowanego z naciskiem na prostotę użytkowania i przejrzystość prezentowania informacji oraz możliwości zapewnionym zalogowanym użytkownikom. |

|  |  |
| --- | --- |
| Wymaganie: | Prostota dostępu do systemu |
| Opis: | Użytkownik powinien być w stanie dostać się do systemu w jak najmniejszej ilości kroków, bez potrzeby uruchamiania na komputerze żadnych zależności. |

## Wymagania dodatkowe

|  |  |
| --- | --- |
| Wymaganie: | Realizacja systemu w postaci aplikacji sieciowej |
| Opis: | System powinien być aplikacją działającą w szpitalnej sieci komputerowej, do której dostęp miałoby wielu użytkowników jednocześnie z własnych komputerów. |

|  |  |
| --- | --- |
| Wymaganie: | Elastyczność wytwarzania systemu |
| Opis: | Proces wytwarzania systemu musiał być elastyczny i przygotowany na zmiany wymagań narzucone przez przedstawicieli zleceniodawcy w trakcie jego implementacji. |

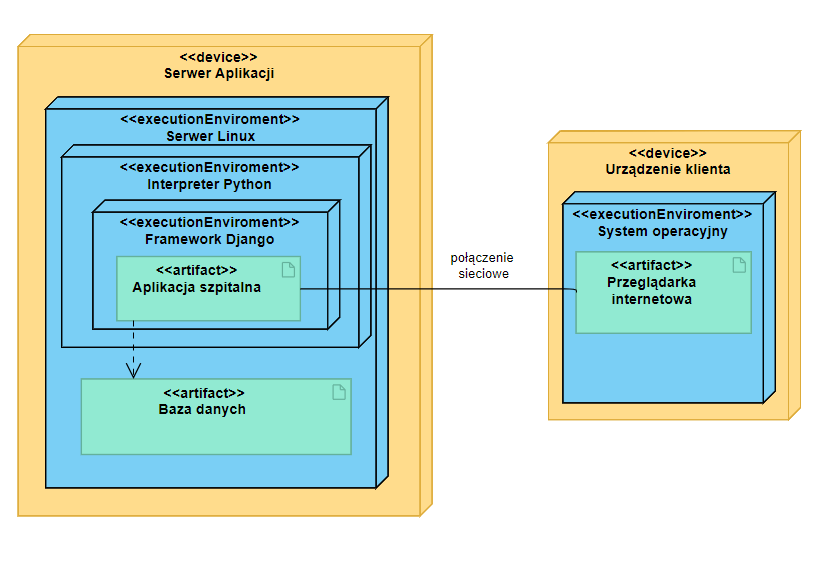
|  |  |
| --- | --- |
| Wymaganie: | Zastosowanie rozwiązań OpenSource |
| Opis: | Z racji przekazania licencji do użytkowania systemu szpitalowi Uniwersyteckiego Centrum Klinicznego nie można wykorzystać płatnych rozwiązań komercyjnych ani rozwiązań darmowych bez gwarancji bezpieczeństwa. |

# Projekt Systemu

## Architektura systemu

Ze względu na wymagania dotyczące realizacji systemu w postaci aplikacji sieciowej oraz wykorzystania technologii OpenSource, zdecydowano zaimplementować go w postaci aplikacji przeglądarkowej we frameworku Django. Przewaga takiego rozwiązania nad aplikacją okienkową wynika z prostoty wdrożenia, gdyż ogranicza się ona jedynie do konfiguracji serwera znajdującego się w szpitalu, bez potrzeby instalowania dodatkowego oprogramowania na wszystkich komputerach lekarzy. Dodatkowo wykorzystując popularne biblioteki stylistyczne CSS możliwe jest zaprojektowanie interfejsu wyglądające znajomo dla użytkowników, co zmniejsza próg wejścia wymagany do biegłego korzystania z systemu osobom niedoświadczonym technicznie zgodnie z wymaganiem dotyczącym prostoty dostępu do systemu.

Na rys. 7 zamieszony został diagram wdrożenia systemu. Elementem podstawowym na nim jest serwer aplikacji z systemem Linux. Jako że framework Django jest napisany w języku Python, wymagana jest obecność jego interpretera na głównym komputerze, aby mógł on służyć jako środowisko wykonawcze dla aplikacji. Dodatkowo na serwerze znajduje się baza danych zawierająca wszystkie informacje do których dostęp zapewnia sama aplikacja. Użytkownicy mogą połączyć się z aplikacją poprzez przeglądarki internetowe zainstalowane na ich służbowych komputerach połączonych ze szpitalną siecią komputerową.



Rys. 7 - Diagram wdrożenia systemu

Warto tutaj wspomnieć o architekturze samego frameworku Django, którą przedstawiono na diagramie na rys. 8. Jest to rozwiązanie o schemacie MVT (Model View Template), będące pewnego rodzaju abstrakcją popularnego MVC (Model View Controler). Część odpowiadająca za reprezentacje modelu jest w nich taka sama i odpowiada ona za komunikację z bazą danych oraz za mapowanie rekordów na obiekty w systemie i spowrotem poprzez system ORM (Object-Relational Mapping). Różnice uwidaczniają się w pozostałych dwóch elementach architektur.

W schemacie MVC widok pełni rolę warstwy prezentacyjnej poprzez prezentowanie użytkownikowi informacji dostarczonych przez model oraz przyjmowanie informacji dostarczonych przez użytkownika. Kontroler odpowiada za komunikację między widokiem i modelem oraz za logikę biznesową aplikacji i generowanie odpowiednich odpowiedzi w zależności od zapytań HTTP wysyłanych przez użytkownika.

W schemacie MVT widok odpowiada za logikę biznesową, przetwarzanie rządań HTTP, interakcje z modelem i dostarczenie szablonowi wybranych danych z modelu. Szablon (ang. Template) jest elementem prezentacyjnym który w całości odpowiada za interfejs użytkownika i konstruowanie dokumentów HTML na podstawie informacji dostarczonych przez widok. W schemacie MVT framework sam pełni rolę kontrolera, oddelegowującego rządania HTTP do odpowiednich widoków.

Można powiedzieć, że MVT jest abstrakcją MVC, gdyż Django i inne frameworki z niego korzystające zawierają w sobie implementacje MVC, jednak jest ona ukryta przed programistą. Warto jednak zaznaczyć, że sami autorzy Django wspominają że rozróżnianie między MVC i MVT może być zagmatwane, mylące, i nie przedstawiające pełnej natury architektury w stu procentach, jako że definicje schematów można interpretować na różne sposoby [11].

Obraz zawierający tekst, urządzenie

Opis wygenerowany automatycznie

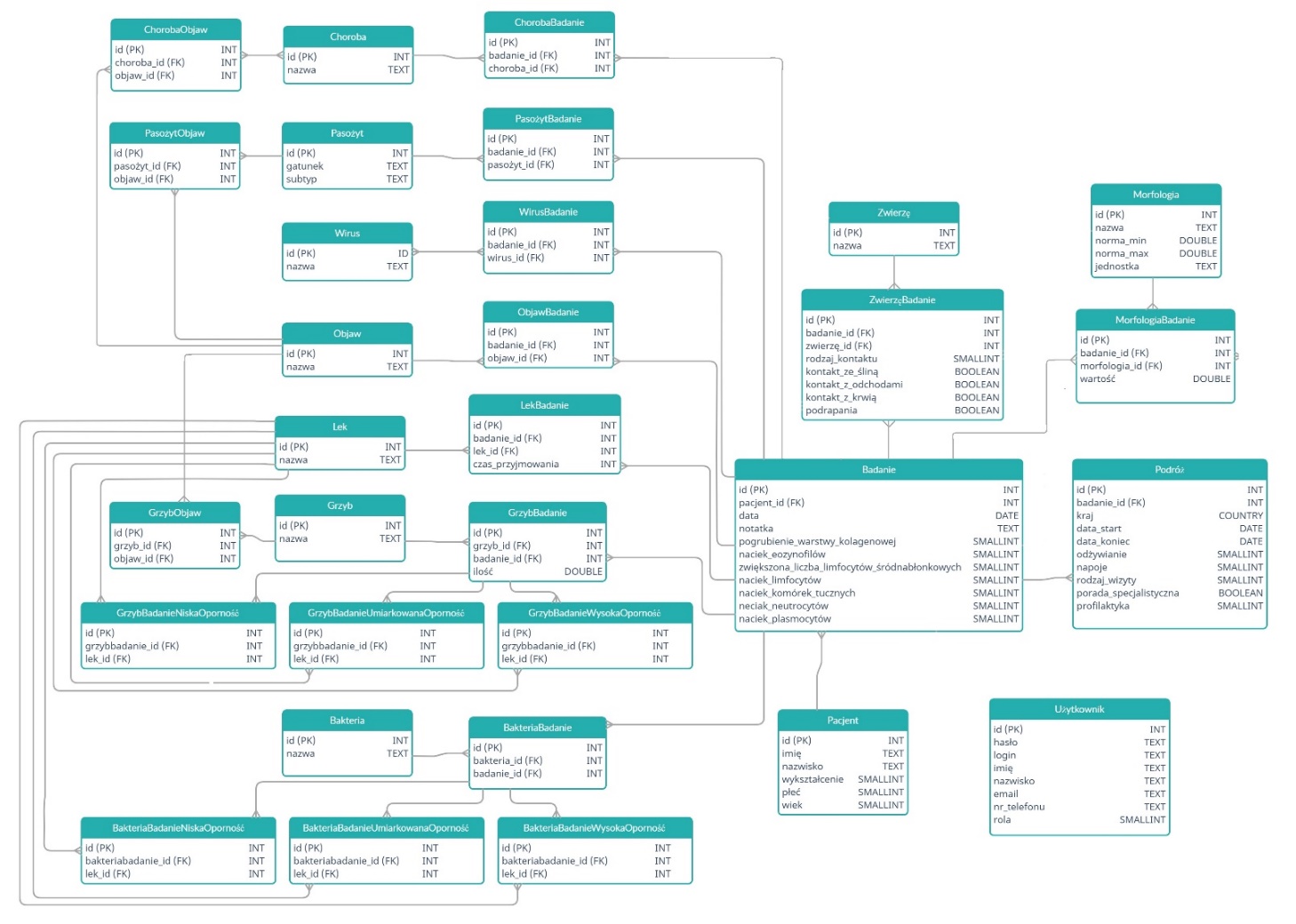
Rys. 8 - Diagram architektury frameworku Django [10]



## Projekt warstwy danych

Jako że głównym celem projektu jest realizacja systemu służącego do analizy relacji między celami podróży pacjentów a różnych patogenów i schorzeń, które u nich wykryto, wybór odpowiedniej technologii baz danych był kluczowym elementem do rozważenia. Biorąc pod uwagę narzucone wymagania mówiące o częstości sprawdzania powiązań między różnymi grupami obiektów, dodawania rekordów przez użytkowników i elastyczności procesu wytwarzania oprogramowania, wykorzystanie relacyjnej bazy danych niesie ze sobą dużo więcej korzyści niż bazy nierelacyjnej. Ostatecznie wybrana została baza PostgreSQL, otwarto-źródłowa technologia opracowana w 1986 roku na Univeristy of California w Berkeley i rozwijana po dziś dzień [12]. Dodatkową zaletą tego rozwiązania w omawianym projekcie jest natywne wsparcie Django dla PostgreSQL, co ułatwiło i skróciło proces implementacyjny.

Na rys. 9 zamieszony został diagram modelu bazy danych. Większość informacji dotyczących wymaganych tablic oraz pól została pozyskana od lekarzy UCK w sposób bezpośredni, lub poprzez analizę dostarczonych formularzy z wywiadów medycznych. Tablice i ich rekordy są mapowane w aplikacji na obiekty pythonowe poprzez system ORM Django. Niektóre z nich zawierają w sobie pola wyboru, które zostały zamodelowane w bazie danych jako pola typu small integer, a ich mapowaniem na wersję czytelną dla człowieka zajmuje się logika biznesowa systemu. Pola te wraz z możliwymi wartościami zostały opisane w tabeli 1.

Rys. 9 - Diagram modelu bazy danych.

|  |  |  |  |
| --- | --- | --- | --- |
| Tablica | Pola | Możliwe wartości | Wersja czytelna dla człowieka |
| Badania | pogrubienie\_warstwy\_kolagenowej, naciek\_eozynofilów, zwiększona\_ilość\_limfocytów\_śródnabłonkowych, naciek\_limfocytów, naciek\_komórek\_tucznych, naciek\_neutrocytów, naciek\_plasmocytów | 0 | Brak |
| 1 | Mało |
| 2 | Umiarkowany |
| 3 | Intensywnie |
| Pacjent | płeć | 0 | Mężczyzna |
| 1 | Kobieta |
| 2 | Inna |
| Pacjent | wykształcenie | 0 | Brak |
| 1 | Podstawowe |
| 2 | Średnie |
| 3 | Wyższe |
| Użytkownik | rola | 0 | Admin |
| 1 | Lekarz |
| 2 | Praktykant |
| Podróż | odżywianie | 0 | Restauracje |
| 1 | Lokalne |
| 2 | Własne |
| 3 | Mieszane |
| Podróż | napoje | 0 | Butelkowane |
| 1 | Lokalne |
| 2 | Strumienie |
| 3 | Mieszane |
| Podróż | rodzaj\_wizyty | 0 | Zawodowy |
| 1 | Turystyczny |
| 2 | Do bliskich |
| Podróż | profilaktyka | 0 | Swoista |
| 1 | Nieswoista |
| Zwierzę | rodzaj\_kontaktu | 0 | Zawodowe |
| 1 | Domowe |
| 2 | Inne |

Tabela 1 – Mapowanie wartości pól wyboru na wartości czytelne dla człowieka

Szczególną uwagę należy zwrócić na tablicę przechowującą dane dotyczące badań pacjentów. Z celu systemu mówiącówiącego o wykorzystaniu go do analizy informacji otrzymywanych z wywiadów medycznych wynika, że rekordy te będą z wysokim prawdopodobieństwem najczęściej pobieranymi i modyfikowanymi. Dodatkowo badania są w relacji z wieloma innymi elementami modelu danych, zazwyczaj o naturze wiele do wielu z tablicami przejściowymi zawierającymi dodatkowe informacje. Z tego powodu należało zastosować specjalne techniki ograniczające nakład na bazę badynch przy obsłudze widoków na nich operujących. Jedną z zastosowanych metod jest stronnicowanie, serwujące klientowi jedynie fragment rekordów w momencie, w którym rząda on ich pobrania. Określenie konkretnego wycinku jest wnioskowane na podstawie parametru zawartego w rządaniu HTTP. Dodatkowym problemem, który należało rozwiązać był problem N + 1 zapytań występujący w większości frameworków wytwarzania aplikacji internetowych. Ukazuje się on w momencie pobierania danych dotyczących pewnych rekordów oraz danych dotyczących N rekordów będących z nimi w relacji. Domyślnym zachowaniem jest wystawienie do bazy danych oddzielnych zapytań dla każdego z nich, co jest dużym nakładem pracy dla systemu DBMS. Programista wiedząc o tym, że takie informacje będą potrzebne, może za wczasu nakazać wczytania tych danych na podstawie informacji o relacji pozyskanych z pierwszego zapytania do bazy danych, co skutkuje jedynie 1 + K zapytaniami, gdzie K jest ilością tablic z którymi pobierane encje są w relacji i na których dane chcemy przedstawić klientowi. Przykład zapytań skonstruowanych na opisane sposoby przedstawiono na rys. 10.

Obraz zawierający tekst, zrzut ekranu

Opis wygenerowany automatycznie

Rys. 10 – Zapytanie do bazy danych pobierające rekordy będące w relacji z rządanymi encjami, skonstruowane domyslnie oraz po zastosowaniu przedwczesnego wczytania.

Kolejnym aspektem, na który należało zwrócić uwagę była natura środowiska szpitalnego w którym system miał być używany. Lekarze powinni mieć dostęp do jak największej ilości informacji dotyczących czynników na które wystawieni byli pacjenci, lecz nie zawsze są one dostępne. Przykładowo pacjent może nie pamiętać lub nie wiedzieć, czy został na wyjeździe podrapany przez zwierzę, z którym miał kontakt, lub nie możliwym może być określenie ilości wykrytego w jego jelitach grzyba. Takie niepełne informacje dalej są wartościowe dla analizy, jednak założenie jakiejś wartości wprowadzałoby zakłamanie do otrzymach wyników. Z tego powodu większość pól w tablicach zezwala na wartość null, aby brakujące informacje nie uniemożliwiały zapisanie takich rekordów.

|  |  |  |
| --- | --- | --- |
| Morfologia | | |
| Pole | Typ | Opis |
| id | INT | Sztuczny klucz główny |
| nazwa | TEXT | Nazwa badania morfologicznego |
| norma\_min | DOUBLE | Minimalna wynik mieszczący się w normie |
| norma\_max | DOUBLE | Maksymalny wynik mieszczący się w normie |

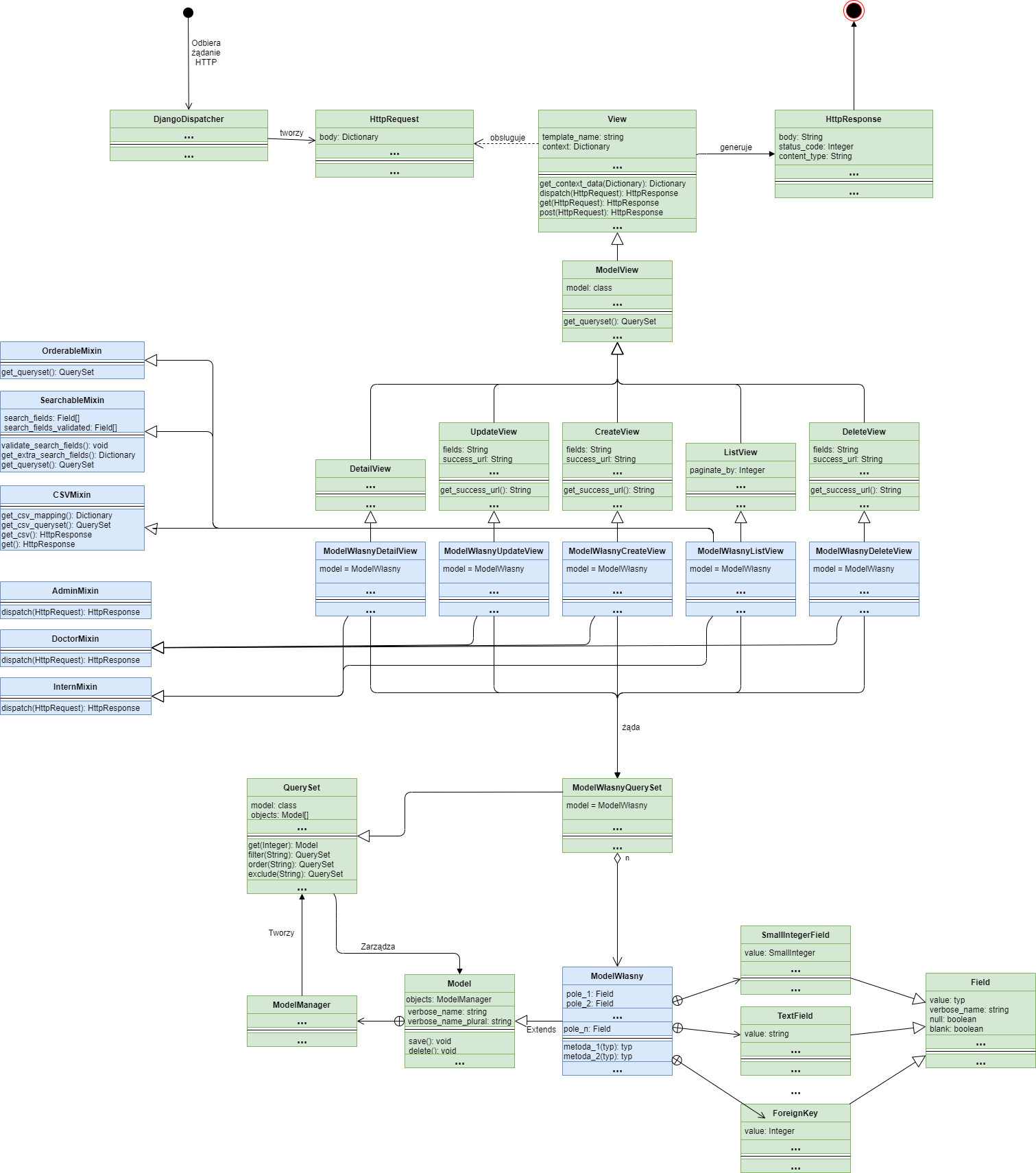
Tabelka X – Gdybym zrobił to dla każdej tablicy to byłby dobry zapychacz, ale większość tablic jest bardzo nieciekawa i oczywista (id z jednej tablicy, id z drugiej tablicy, nazwa = nazwa obiektu)

## Projekt warstwy logiki biznesowej

Ze względu na wykorzystanie frameworka Django do realizacji systemu model logiki biznesowej został w dużej mierze wyzanczony przez jego autorów. Na rysunku nr 11 przedstawiony został uproszczony diagram modelu klas z punktu widzenia logiki biznesowej. Implementacja własnych klas w systemie była oczywiście niezbędna do osiągnięcia wymaganych efektów, jednak nie można uznać, że którakolwiek z nich istnieje w systemie w izolacji od dostarczonych elementów. Ich dokładny opis musiałby być jednak bardzo obszerny, a ostatecznie nie mówiłby wiele o esencji samego systemu w kontekście jego roli biznesowej. Dodatkowo, funkcjonalności zarządzania poszczególnymi modelami w systemie oraz sposoby ich działania są niemalże identyczne, przez co różnice między nimi wynikają jedynie z rodzajów pól znajdujących się w tablicach reprezentujących je w bazie danych. Z tych powodów postanowiono zastosować pewne założenia przy przedstawieniu diagramu klas:

* Klasy dostarczone przez Django zostały przedstawione na diagramie w kolorze zielonym, zaś klasy zaimplementowane przez autora systemu w kolorze niebieskim,
* Pola oraz metody zawierające się w klasach dostarczanych przez Django są przedstawiane wyłącznie w sposób wyrywkowy i uproszczony wedle zasady czarnej skrzynki, tak aby przedstawić tylko te elementy które są kluczowe dla działania systemu i komponentów autorskich z nimi współpracującymi. Dokładny opis działania i dziedziczenia klas wbudowanych znajduje się w dokumentacji Django [13]
* Na diagramie nie przedstawiono żadnej konkretnej klasy autorskiej, lecz pewną hipotetyczną klasę, za którą można by było wstawić klasy z dowolnego modułu w systemie.

Diagram dla ułatwienia jego zrozumienia można podzielić na dwie części: opisującą klasy opowiadające za komunikację z modelem danych i systemem ORM, oraz na część opisującą klasy widoków, odpowiedzialnych za obsługę żądań HTTP przesyłanych przez użytkownika i generowanie odpowiedzi.



Rys. 11 Uproszczony model klas logiki biznesowej.

## Klasy ORM

Najważniejszym bytem dostarczanym przez Django do celu zaimplementowania modelu danych jest abstrakcyjna klasa Model. Pozwala ona na odwzorowanie jej przedków, to znaczy klas po niej dziedziczących, na poszczególne tablice i wiersze w bazie danych poprzez skojarzenie klasy z konretną tablicą oraz dostarczenie metod pozwalających na wykonywanie operacji CRUD, czyli tworzenia, pobrania, modyfikacji oraz usunięcia obiektu z bazy danych. Przykładowo, aby pozwolić systemowi na operowanie na bakteriach zawartych w bazie danych, koniecznym jest utworzenie klasy Bakteria, która dziedziczyłaby po klasie Model. Do zareprezentowania poszczególnych kolumn z tablicy służą dostarczone klasy będące uszczególnieniem abstrakcyjnej klasy Field, takie jak TextField, IntegerField, czy ForeignKey. Są one odpowiedzialne nie tylko za zdefiniowanie odpowiednich kolumn w danej tablicy bazodanowej, ale także za walidacje dostarczonych danych pod względem typu lub innych określonych parametrów (jak na przykład wartości maksymalne, lub zezwolenie na wartość null), oraz za sposób przedstawienia kolumny użytkownikom systemu w sposób czytelny dla człowieka. Dodatkowo programista może wskazać wybrane pola i nakazać ich zaindeksowania w celu zoptymalizowania operacji bazodanowych.

Jednym z kluczowych pól Modelu jest pole „objects” klasy ModelManager, pozwalające na utworzenie nowego obiektu typu QuerySet danej klasy. Do każdego modelu automatycznie określana jest oddzielna klasa dziedzicząca po klasie QuerySet. Przykładowo pole „objects” klasy Bakteria daje dostęp do nowego obiektu klasy BakteriaQuerySet. Klasa ta pozwala na konstruowanie złożonych zapytań do bazy danych, poprzez dostarczenie metod pozwalających na pobieranie poszczególnych wierszy na podstawie wybranych pól, filtrowanie wyników na podstawie określonych parametrów, określenie konkretnych kolumn do wczytania, czy też oznaczenie pól zawierających w sobie klucze obce do przedwczesnego wczytania wierszy z tablic będących w relacji z obiektami pobieranymi (w celu zapobiegnięcia problemu N+1 zapytań). Jeżeli konkretny przypadek użycia byłby zbyt skomplikowany, to możliwym jest też sporządzenie surowego zapytania SQL z poziomu kodu samej aplikacji.

Ważną cechą klay QuerySet jest jej leniwość, polegająca na odwleczeniu wysłania zapytania do bazy danych do momentu, kiedy żądane obiekty są wymagane przez któryś z elementów systemu. Oznacza to, że możliwym jest łańcuchowanie poszczególnych metod klasy QuerySet oraz przetrzymywanie określonych zapytań w systemie, a następnie wysłanie pojedynczego, rozbudowanego zapytania SQL w momencie, w którym dane są potrzebne, co znacząco zmniejsza nakład pracy narzucany na system DBMS.

## Klasy widoków

W momencie, w którym serwer Django odbiera od użytkownika żądanie HTTP framework sam odpowiada za zdelegowanie jego obsługi do odpowiednich widoków na podstawie mapy adresów URL. Jest to zgodne ze schematem MVT który został opisany w rozdziale 4.1 - Architektura systemu. Autorzy Django dostarczyli gotowe klasy abstrakcyjne reprezentujące widoki w sposób bardzo ogólny, jak i bardziej sprecyzowane na podstawie najpopularniejszych zastosowań. Mowa tutaj o widokach odpowiedzialnych za tworzenie, modyfikowanie, usuwanie, listowanie lub pobranie konkretych obiektów danej klasy modelu. Odpowiedzialność za dopasowanie tych widoków do konkretnych modeli wraz z dostarczeniem dodatkowych funkcjonalności pada jednak już na programistę. Przykładowo, aby pozwolić użytkownikowi na tworzenie nowych obiektów typu Bakteria i zapisanie ich do bazy danych zaimplementowana została klasa BacteriaCreateView dziedzicząca po klasie Django CreateView. Klasy dostarczone przez framework posiadają już zaimplementowane operacje pozwalające na obsługę żądań, konstruowanie plików HTML na podstawie określonych szkieletów, stronnicowanie list obiektów, czy też pobranie określonych danych poprzez wywołanie nowych obiektów klasy QuerySet, chociaż zastosowanie bardziej wykwintnych filtrów spada już na barki programisty.

Widoki dostarczane przez Django są dobrą podstawą do obsługi żądań, jednak wymagania systemu narzucone przez przedstawicieli UCK wymagały bardziej złożonych mechanizmów niż te zapewnione. W tym celu zaimplementowane zostały klasy pomocnicze, które rozszerzają pewne domyslne metody widoków oraz wstrzykują dodatkowe funkcjonalności. Jako że jednym z podstawowych wymagań jest możliwości przeprowadzania szczegółowej analizy danych, to podstawowe działanie widoków listowych polegające na pobraniu wszystkich obiektów danego modelu jest zbyt prymitywne. Do rozszerzenia możliwości dostarczonych użytkownikom służą poszczególne klasy:

* OrderableMixin – Dostarcza widokom możliwość sortowania wyników po kolumnach sprecyzowanych w parametrach URL żądania HTTP,
* SearchableMixin – Dostarcza widokom możliwość filtrowania wyników na podstawie parametrów URL żądania HTTP. Domyślne działanie opiera się na wartościach określonych pól, jednak pozwala na zaimplementowanie i doczepienie bardziej wyszukanych filtrów w poszczególnych widokach,
* CSVMixin – Umożliwia przesłanie odpowiedzi HTTP w postaci pliku CSV zawierającego wszystkie wystawione obiekty.

Użytkownik może doczepić takie parametry URL przy pomocy interfejsu graficznego, co zostało opisane dokładniej w późniejszym rozdziale. Funkcjonalności te zostały oddelegowane do własnych klas, gdyż przeprowadzenie tych operacji wymagane było dla każdej klasy modelu. Pozwoliło to na modularne implementowanie widoków listowych i zachowanie zasady DRY (ang. Don’t Repeat Yourself), co przyczyniło to się do efektywniejszego wytwarzania oprogramowania oraz lepszej otwartości na modyfikacje regularnie narzucane przez lekarzy.

Pozostałe trzy klasy pomocnicze związane są kontrolą dostępu dla różnych typów użytkowników, tak jak sprezyzowane to zostało w wymaganiach systemowych. Są to kolejno:

* InternMixin – widoki upublicznione dla stażystów pracujących w szpitalu. Posiadają oni dostęp do widoków listowych i do widoków przedstawiających detale konkretnego obiektu. Nie posiadają oni możliwości modyfikowania obietków znajdujących się w bazie danych
* DoctorMixin – widoki upublicznione dla regularnych lekarzy pracujących w szpitalu. Posiadają oni dostęp do wszystkich widoków do których dostęp mają stażyści, oraz do widoków odpowiedzialnych za tworzenie, modyfikowanie oraz usuwanie obiektów.
* AdminMixin – widoki upublicznione dla administratorów systemu. Posiadają oni dostęp do każdego widoku w systemie, to znaczy do tych widoków, do których dostęp posiadają lekarze, oraz do widoków pozwalających na zarządzanie użytkownikami systemu.

W przypadku, w którym użytkownik anonimowy (niezalogowany) próbowałby otrzymać od serwera jakiekolwiek informacje, zostałby on przekierowany na widok logowania. Jeżeli użytkownik zalogowany próbowałby dostać się do części aplikacji do której nie jest upoważniony, zostałby o tym poinformowany i przekierowany na stronę startową aplikacji.

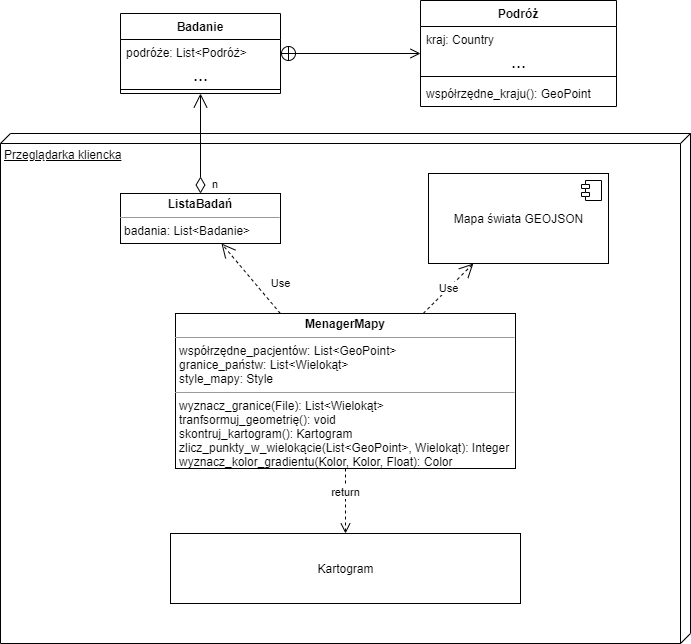
## Obsługa badań i wywiadów medycznych

Większość modeli i klas w systemie można sprowadzić do schematu przedstawionego w poprzednim podrozdziale, jednak z powodu wymogu zestawienia informacji dotyczących pacjentów na mapie świata proces ten jest bardziej złożony dla obsługi badań i wywiadów medycznych. O ile proces przetwarzania żadania HTTP po stronie serwera przebiega dokładnie tak samo jak w przypadku innych klas w systemie, tak po stronie klienta zaimplementowana została dodatkowa logika odpowiadająca za konstrukcję odpowiedniego kartogramu. Została ona przedstawiona na diagramie na rys. 11. Pierwszym etapiem przetwarzania jest wyodrębnienie informacji o państwach odwiedzonych przez wybranych pacjentów (poddanych ewentualnym operacjom filtrujących) i przekonwetrowanie ich na odpowiednie współrzędne geograficzne. Jako że do konstrukcji kartogramu nie potrzeba jest dokładna lokalizacja próbek badawczych, a jedynie informacja o tym w których obszarach się znajdują, postanowiono dokonoć konwersji państw odwiedzanych na współrzędne geograficzne ich stolic.

Drugim elementem potrzebnym do skonstruwania kartogramu jest informacja o granicach nałożonych na mapę świata. Do tego celu zastosowano mapę świata w formacie GEOJSON. Jest ona zareprezentowana jako lista obiektów w formacie JSON reprezentujących poszczególne państwa. Obiekt taki składa się z pól zawierających różne informacje dotyczące danego kraju, takie jak jego nazwa, rozmiar populacji, sugerowane style przy renderowaniu mapy oraz jego granice. Te ostatnie z kolei są przedstawione jako lista poszczególnych terytoriów, które przedstawione są jako listy punktów geograficznych, których połączenie linią na mapie formuje wielokąt. Jako że mapa identyfikuje państwa jako rozróżnialne byty, koniecznym jest lista terytoriów do wyznaczenia jego granic, gdyż wiele z nich położonych jest na niejednolitych terenach rozdzielonych morzami lub oceanami, np. Japonia składająca się z wielu wysp, lub Francja i jej terytoria zamorskie (przykładowo Gujana Francuska), jednak dalej stanowią one tą samą jednostę administracyjną. Ważnym parametrem map GEOJSON jest ich rozdzielczość, mówiąca o tym, ile powinna wynosić maksymalna odległość pomiędzy poszczególnymi punktami określającymi granice terytoriów. W internecie można znaleźć mapy o rozdzielczości od kilku do kilkuset metrów. Im mapa jest dokładniejsza, tym lepiej przedstwione granice odzwierciedlają rzeczywisty stan rzeczy, ale kosztem jest ilość punktów potrzebnych do ich przedstawienia, co z kolei przekłada się na większy rozmiar pliku GEOJSON oraz na większą ilość obliczeń wymaganych do wyznaczenia punktów zawierających się w danym wielokącie, co jest operacją konieczną do skonstruowania kartogramu. Jako że do analizy danych dotyczących pacjentów wystarczający jest pogląd globalny oraz biorąc pod uwagę fakt, że wszystkie współrzędne rzutowane są na stolice danych państw, postanowiono w systemie wykorzystać mapę o niskiej rozdzielczości o dokładności do 110 metrów.

Kolejną kwestią do roztrzygnięcia związaną z wykorzystaniem mapy GEOJSON był zastosowany przez jej autorów podział na jednostki administracyjne. Wiele map o niskiej rozdzielczości traktuje państwa posiadające terytoria zamorskie jako jeden obszar administracyjny, lub nie zawiera w sobie terytorów małych wysp oddalonych od kontynetów. Jako że główną grupą badawczą którą zainteresowani są lekarze z UCK byli pacjenci powracający z krajów egzotycznych, stanowiło to istotny problem. Osoby podróżujące do terytoriów zamorskich, takich jak na przykład Gujana Francuska czy Reunion, wpływałaby na swoje statystyki, tak jak i na statystyki samej Francji. Jednocześnie małe terytoria często są popularnymi celami turystycznymi i brak ich reprezentacji na mapie GEOJSON uniemożliwiłby analizę danych pacjentów do nich podróżujących. Po drugiej stronie ekstremum są mapy, które stosują zbyt szczegółowy podział na poszczególne jednostki administracyjne, takie jak województwa w Polsce, stany w USA, czy hrabstwa w Wielkiej Brytanii. Nie dość, że mapy takie muszą cechować się większą dokładnością, czego minusy zostały omówione powyżej, to dodatkowo nie da się skonstruować na ich podstawie kartogramu obejmującego poszczególne państwa. Przykładowo pacjenci powracający z Japonii zostaliby przedstawieni na mapie świata jedynie w prefekturze Kentou, jako że to właśnie tam znajduje się Tokyo będące stolicą tego kraju (do której sprowadzane są współrzędne całej próbki), zaś pozostałe obszary państwa pozostałyby niezakolorowane. Ostatecznie postanowiono wykorzystać w systemie mapę o niskiej rozdzielczości, zawierającej większość małych terytoriów niezależnie od ich odległości od kontynentu oraz z podziałem administracyjnym łączącym wszystkie terytoria danego państwa jako jeden byt. Nie było to rozwiązanie idealne, jednak biorąc pod uwagę ograniczoną ilość zasobów w internecie był to najbardziej akceptowalny kompromis.

Na podstawie wybranej mapy GEOJSON oraz listy współrzędnych geograficznych państw odwiedzonych przed pacjentów możliwe jest skonstruowanie kartogramu. Najpierw należy jednak sprowadzić współrzędne podróży oraz współrzędne granic do wspólnego formatu projekcji na mapie. W systemie wykorzystywana do tego jest projekcja EPSG:4326, będąca najbardziej popularną w typowych zastosowaniach geoinformatycznych. Następnie dla każdego obszaru na mapie zliczane są punkty zawierające się w nich i wyznaczany jest ich stosunek względem wszystkich punktów w celu dobrania odpowiedniego koloru gradientu. Do tego celu zastosowano bibliotekę języka JavaScript o nazwie JSTS (będącą portem biblioteki JTS języka Java) dostarczającą funkcje służące do przetwarzania informacji geograficznych. Ostatnim krokiem jest wyrenderowanie gotowego kartogramu. Do tego celu wykorzystana została biblioteka OpenLayers, służąca do osadzania różnego rodzaju interaktywnych map do dokumentu HTML.



Rys. 12 Diagram logiki biznesowej renderowania kartogramu.

## Projekt interfejsu użytkownika

Jednym z głównych wymagań systemowych był prosty i intuicyjny interfejs użytkownika, z którego korzystać z łatwością mogłyby osoby niedoświadczone technicznie. Z tego powodu graficzny interfejs był koniecznością, gdyż korzystanie z konsoli i ręcznego wpisywania komand w celach interakcji z systemem byłoby procedurą zbyt skomplikowaną. Decyzja o realizacji systemu w postaci aplikacji przeglądarkowej była także kierowana tym powodem, gdyż większość ludzi jest zaznajomiona z nawigowaniem stron internetowych, więc wykorzystanie podobnego układu stron do tych spotykanych na popularnych serwisach zmniejszyłoby próg wejścia do biegłego korzystania z systemu.

Jako że większa część systemu pełni rolę interfejsu pozwalającego na operowanie na bazie danych poprzez wykonywanie podstawowych operacji CRUD oraz filtrowania wyników, to podstrony odpowiadające za poszczególne modele zostały skonstruowane na podstawie jednego szkieletu dopasowanego pod konkretną klasę. Każda taka podstrona zawiera w sobie widok wylistywujacy wiersze pobrane z bazy danych, widok pozwalający na wprowadzenie nowych rekordów, widok pozwalający na usunięcie wybranego rekordu, widok pozwalający na edycję wybranego rekordu oraz wyidok pozwalający na wyświetlenie szczegółowych informacji o danej encji, jeżli są one zbyt bogate, aby przedstawić je w sposób przejrzysty na liście.

Na górze strony znajduje się menu nawigacyjne pozwalające użytkownikowi przejść na podstronę konkretnego modelu. Dodatkowo znajdują się tam przejścia prowadzące na stronę pozwalającą zobaczyć i zedytować dane profeilowe użytkownika oraz przycisk wylogowujący, lub w przypadku uzytkowników niezalogowanych przejście do strony logowania. Każde przejście na inną podstronę jest poprzedzone animacją kręcącego się okręgu, aby zasygnalizować użytkownikowi, że strona się ładuje. W przeciwnym wypadku mógłby on nie wiedzieć że serwer obsługuje aktualnie jego żądanie i pomyśleć, że strona się zawiesiła, lub że nie wykonał operacji która w rzeczywistości została odebrana. Do stylizacji strony postanowiono wykorzystać framework stylistyczny CSS o nazwie Bootstrap w wersji 4. Jest to rozwiązanie otwartoźródłowe wykorzystywane przez wiele znanych serwisów internetowych, takich jak Spotify czy Twitter, co zwiększa szanse na to, że użytkownik miał już wcześniej doświadczenie w nawigowaniu po podobnych stronach. Drugim powodem wykorzystania tego rozwiązania był fakt, że przyspieszyło ono proces wytwarzania oprogramowania.

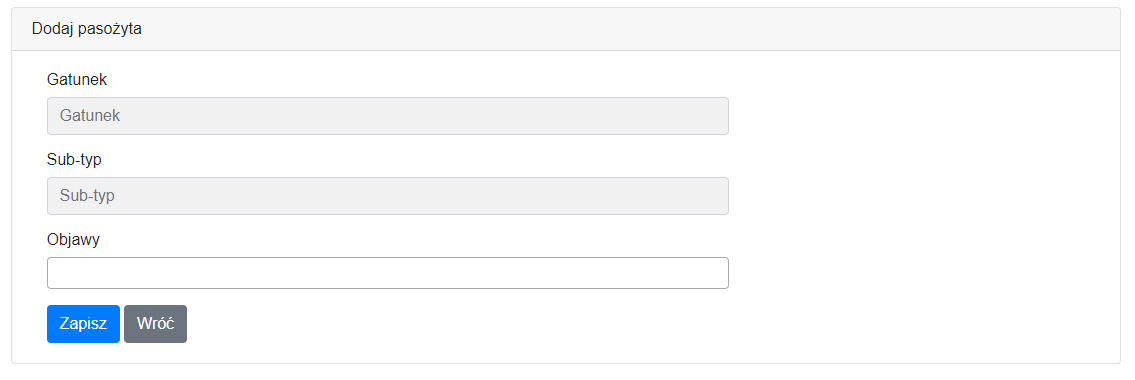
Głównym celem widoków listowych jest przedstawienie użytkownikowi wybranych rekordów z bazy danych dotyczących konkretnego modelu. Zrzut ekranu takiego widoku dla badań laboratoryjnych (pierowotnie nazwanych morfologie – zmienione na późniejsze życzenie klienta) zamieszczony został na rys. 13. Rekordy umieszczone w tabeli znajdującej się w centrum strony. Oprócz wybranych kolumn, przy każdym wierszu znajdują się przyciski pozwalające na usunięcie oraz edycję lub podgląd danego elementu, jeśli wszystkie jego pola nie mogłyby być przedstawione na jednym ekranie w sposób estetyczny. Przyciski te są ukryte dla użytkowników nieuprawnionych, aby nie przytłaczać go informacjami i opcjami z których i tak nie może skorzystać. Wyniki można sortować rosnąco lub malejąco na podstawie wybranych kolumn poprzez kliknięcie nagłówków oznaczonych strzałką. Kierunek strzałki informuje o tym, czy sortowanie po kolumnie jest w danym momencie aktywne oraz w przypadku w którym jest, czy jest to sortowanie rosnące czy malejące. Nie każda kolumna podlega sortowaniu, gdyż nie ma na to potrzeby biznesowej (np. nie ma potrzeby sortowania badań laboratoryjnych po ich normach), lub nie ma oczywistego sposobu w jaki można porównać nimi różne rekordu (np. pola wiele do wielu). Nad tabelką znajduje się formularz służący do filtrowania wyników na podstawie określonych parametrów. Użytkownik może zastosować filtry poprzez wciśnięcie klawisza Enter na klawiaturze lub poprzez wciśnięcie przycisku „Szukaj” znajdującego się pod formularzem. Na prawo od niego znajdują się przyciski pozwalające na wyczyszczenie formularza i ponownym wyświetleniu wszystkich wyników oraz przycisk służący do pobrania aktualnie wyświetlanych rekordów w postaci pliku CSV. Pod tabelką znajduje się lista stron pozwalająca na wyświetlenie dodatkowych rekordów oraz informująca użytkownika o tym na której stronie aktualnie się znajduje poprzez zakolorowanie numeru bierzącej strony. Na samej górze widoku zamieszczona jest nazwa aktualnie obsługiwanego modelu napisana wielkimi literami, aby użytkownik cały czas widział w jakiej części aplikacji się znajduje. Pod nagłówkiem znajduje się przycisk pozwalający na dodanie nowego rekordu, jeżeli użytkownik posiada do tego uprawnienia.

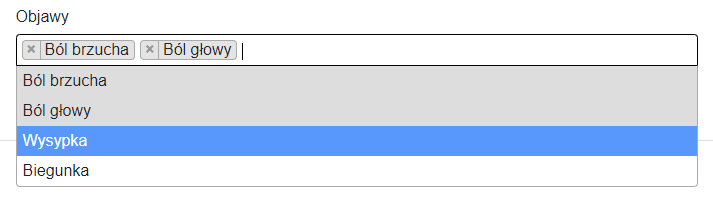
Obraz zawierający stół

Opis wygenerowany automatycznie

Rys. 13 Widok listowy (Badania laboratoryjne).

Po przejściu na widok listowy wyświetlany jest formularz pozwalający na dodanie nowego rekordu do konkretnej tablicy w bazie danych, przedstawiony na rys. 14. Konkretne pola umieszczone są kolejno pod sobą, wraz z nagłówkami informującymi użytkownika czego tyczy się dana formatka. Większość pól wprowadzana jest w klasyczny sposób, czyli poprzez wpisanie odpowiedniej wartości. Wyjątkami od tej reguły są pola zawierające klucze obce lub daty. Aby użytkownik nie musiał znać i pamiętać konkretnych kluczy głównych obiektów które mają być w relacji z rekordem tworzonym, zastosowano formatki Select2 [16]. Pozwalają one na łączenie i rozłączanie obiektów w sposób intuicyjny, bez potrzeby ręcznego wpisywania żadnych informacji. Przykład formatki zamieszczony został na rys. 15. Jeżeli relacja posiada dodatkowe informacje o swojej naturze, zastosowano w formularzu karty, pozwalające na dynamiczne dodawanie i odejmowanie pól, czego przykład przestawiono na rys. 16. W przypadku pól zawierających w sobie datę wykorzystano interaktywny kalendarz (rys. 17), w którym użytkownik musi jedynie wcisnąć datę, którą chce wprowadzić. W przypadku, w którym użytkownik próbowałby zapisać nowy rekord, ale wprowadzone dane nie przeszłyby procesu walidacji, formularz wyświetli komunikat mówiący o wykrytych błędach, co przedstawiono na rys. 18. Widok edycyjny korzysta z tych samych dokumentów HTML co widok dodawania rekordu, ale uzupełnia poszczególne pola wartościami znajdującymi się w wybranym rekordzie. Po wprowadzeniu wszystkich zmian użytkownik jest przekierowywany na widok listowy.



Rys. 14 Widok dodawania nowego rekordu (Badania laboratoryjne).  


Rys. 15 Formatka Select2 służąca do wprowadzania kluczy obcych.  
Obraz zawierający tekst, zrzut ekranu, wewnątrz

Opis wygenerowany automatycznie

Rys. 16 Karty pozwalające na dynamiczne dodawanie i odejmowanie pól z formularza.

Obraz zawierający tekst

Opis wygenerowany automatycznie

Rys. 17 Pole wprowadzania daty  
Obraz zawierający tekst

Opis wygenerowany automatycznie

Rys. 18 Błąd w formularzu

Jeżeli dany model jest bardzo obszerny (np. wywiady medyczne i badania pacjenta), przejście na widok edycji w tabeli zamieniony został zamieniony na widok szczegółowy, przedstawiający wszystkie informacje tyczące się wybranego rekordu w postaci wypełnionych pól i tabelek. Jako że większość tablic w bazie danych zezwala na wartości typu null, które biznesowo przedstawiają wartości nieznane, to wartości takie zostały przedstawione jako trzy znaki zapytania. Przykład takiego widoku dla wywiadów medycznych i badań został przedstawiony na rys. 19. Na samym dole widoku edycyjnego znajdują się przyciski „Wróć” oraz „Edytuj”, pozwalające kolejno wrócić do widoku listowego oraz przejść do widoku edycyjnego.

Obraz zawierający tekst

Opis wygenerowany automatycznie

Rys. 19 Fragment widoku szczegółowego.

Wciśnięcie przycisku „Usuń” na widoku listowym przekierowuje użytkownika na widok usuwania, na którym jest on poproszony o potwierdzenie swojej decyzji. Służy to zapobiegnięciu przypadkowego usunięcia któregoś z rekordów. Widok został przedstawiony na rys. 20.

Obraz zawierający tekst

Opis wygenerowany automatycznie

Rys. 20 Widok potwierdzenia usunięcia rekordu.

Ze względu na rozszerzoną logikę biznesową widok listowy wywiadów medycznych i badań pacjntów jest bardziej rozbudowany od pozostałych. Po pierwsze zawiera bardziej rozbudowany formularz filtrowania wyników który zawiera w sobie także pola ugrupowone, to znaczy takie, które wymagają wypełnienia kilku pól w celu zastosowania jednego filtra. Przykładowo, możliwym jest wybranie tylko takich rekordów, którę posiadają konkretny rodzaj grzyba wykazującego dany stopień oporności na specyficzny lek. Drugim, zapewne najistotniejszym elementem, jest mapa świata w postaci kartogramu. Obok przycisku „Dodaj” znajdującego się na górnej częsci strony, zamieszczony jest przycisk „Pokaż mapę” po którego wciśnięciu zostanie ona wyrenderowana nad formularzem filtrującym. Mapę postanowiono ukryć za przyciskiem, ponieważ jej wygenerowanie może być operacją czaso chłonną, co wydłużyłoby czas ładowania strony. Dodatkowo mapa zajmuje większość ekranu, co mogłoby wprowadzić użytkownika chcącego przejrzeć dostępne rekordy w zamieszanie. Mapa jest interaktywna, co oznacza, że możliwym jest jej przesuwanie oraz przybliżanie i oddalanie. Każdemu kraju na niej przypisane są odpowiednie kolory mówiące o stopniu, w którym były one odwiedzane przez wybranych pacjentów, gdzie kraje najmniej odwiedzane zaznaczone są kolorem zielonym, zaś te najczęściej odwiedzane kolorem czerwonym. Próbki pacjentów na podstawie których kartogram jest rysowany wyznaczane są na podstawie filtrów zastosowanych na aktualnym widoku. Kartogram wraz z przyciskiem go aktywującym został zamieszczony na rys. 21.

Obraz zawierający mapa

Opis wygenerowany automatycznie

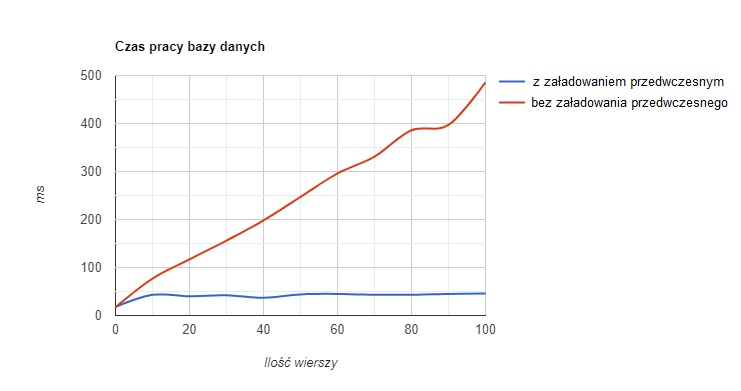
Rys. 21 Kartogram przedstawiający miejsca występowanie wykrytych skorzeń i patogenów wykrytych u pacjentów.

# Testy systemu

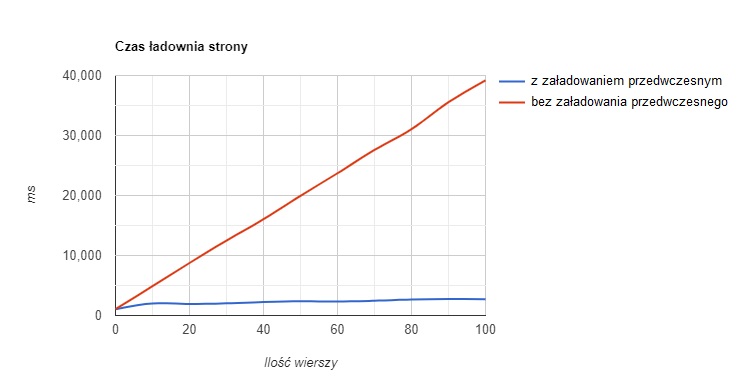
Zważając na rozległość systemu wynikającą z liczności poszczególnych modułów, problematycznym jest wyznaczenie najlepszej metryki do oceny jakości działania systemu. Z tego powodu postanowiono skupić się na przebadaniu czasu ładowania podstrony listującej rekordy wywiadów medycznych i badań pacjentów przy zastosowaniu różnych opcji konfigiguracyjnych. Było to uargumentowane faktem, że widok ten jest najbardziej złożoną częścią aplikacji oraz jest on uzależniony od reszty systemu, a dodatkowo jest on bardziej złożony od innych podstron ze względu na zaimplementowaną w nim funkcjonalność przedstawiania wyników na mapie świata. Postanowiono skupić się na testach wydajnościowych w zależności od ilości adekwatnych rekordów w bazie danych oraz wpływu mechanizmów przedwczesnego zaciągania wyników powiązanych, stronnicowania i rozdzielczości mapy GEOJSON na tęże wydajność. Omówiono takze jak wyglądał proces testów akceptacyjnych u klienta i jaki wpływ miały one na proces wytwarzania oprogramowania.

## Testy wydajnościowe

Pierwszą zastosowaną techniką optymalizacyjną było przedwczesne zaciągnięcie wyników powiązanych z wywiadami medycznymi i badaniami pacjentów. Zwracając uwagę na fakt, że przedstawienie wyników użytkownikowi w postaci tabelki wymaga wyświetlenia informacji o wszystkich pozostałych częściach systemu (takich jak wykryte u pacjenta grzyby, pasożyty, jego kontakty ze zwierzętami, cele podróży i inne), to wyraźnie uwidaczniał się potencjał optymalizacyjny wynikacjący z redukcji ilości zapytań SQL do bazy danych. Czas pracy bazy danych przed i po zastosowaniu przedwczesnego ładowania wyników powiązanych został przedstawiony na wykresie na rys. 22. Można na nim zauważyć, że domyślnie czas pracy bazy danych rośnie w sposób zbliżony do liniowego w zależności od ilości wierszy w bazie danych reprezentujących poszczególne wywiady medyczne, zaś w przypadku zastosowania załadowania przedwczesnego czas pracy bazy danych wzrasta w stopniu dużo mniejszym. Były to wyniki spodziewane, zwracając uwagę na fakt, że wywiady medyczne są w relacji z 11 innymi modelami, co domyślnie przekłada się na tyle samo zapytań dla każdego pobranego rekordu wywiadów medycznych. Przy zastosowaniu załadowaniu przedwczesnym z kolei ilość zapytań skierowanych do bazy danych jest stała i ewentualne różnice czasowe wynikają jedynie z rozmiarów tablic, które system DBMS musi przesłać spowrotem do serwera. Mimo że różnica czasu pracy bazy danych na komputerze, na którym przeprowadzano testy mieściła się w przedziela kilkuset milisekund, to przekładało się to na dzesiątki sekund różnicy w czasie ładowania się całej strony, co przedstawiono na wykresie zamieszczonym na rys. 23. Najprawdopodoniej wynikało to z nakładu pracy narzuconego na serwer wiążącego się z koniecznością zainicjowania każdego zapytania z osobna oraz otworzenia dla niego połączenia z bazą danych, co przy takiej ilości powtórzeń jest czynnością bardzo czasochłonną.

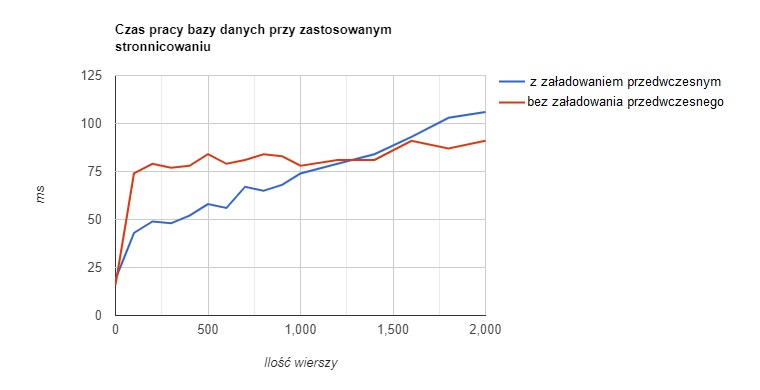


Rys. 22 Czas pracy bazy danych w zależności od zastosowania mechanizmu przedwczesnego zaciągania wyników powiązanych.



Rys. 23 Czas ładownia strony w zależności od zastosowania mechanizmu przedwczesnego zaciągania wyników powiązanych.

Patrząc na powyższe wyniki można wyjść z założenia, że przedwczesne ładowanie rekordów powiązanych jest mechanizmem, który warto zastosować w każdym wypadku, kiedy aplikacja wykorzystuje modele będące między sobą w relacji. Jest to jednak podejście naiwne, co uwidacznia się przy większej ilości rekordów znajdujących się w bazie danych oraz przy zastosowaniu mechanizmu stronnicowania. W takim wypadku czas pracy bazy danych jest większy dla mniejszej ilości wyników bez załadowania przedwczesnego, jednak wzrasta on w większym tempie, kiedy ten mechanizm jest zastosowany, co przedstawiono na wykresie na rys. 24. Do wyjaśnienia tych wyników kluczowym jest zrozumienie na czym dokładnie polega problem N+1 zapytań i zauważenie, że występuje on tylko w przypadku, w którym aplikacja musi pobrać informacje zawierające się w poszczególnych kolumnach z tablic będących w relacji z modelem głównym. W momencie jednak w którym zastosowane zostało stronnicowanie, strona przedstawia użytkownikowi w danym momencie jedynie kilka rekordów z całej puli, niezależnie od jej rozmiarów (w trakcie testowania było to 10 stron), chociaż ze względu na zamieszczenie mapy świata na badanym widoku dalej wymagane było pobranie informacji o wszystkich celach turystycznych pacjentów. Dla mniejsze całkowitej ilości rekordów ilość zapytań wynikająca z ilości wyświetlanych wierszy bez załadownia przedwczesnego była czynnikiem bardziej kosztownym czasowo. Jednak wraz z wzrostem ilości wszystkich wierszy koszt wynikający z zaciągania wszystkich pokrewnych rekordów okazał się być bardziej znaczący niż nakład wynikający ze stałej ilości zapytań SQL przy zachowaniu domyślnym. Po raz kolejny drastyczną różnicę czasową dało radę zauważyć w całkowitym czasie ładowania strony, który był początkowo większy bez załadowania przedwczesnego, jednak jego tempo wzrostu było szybsze przy zastosowaniu tego mechanizmu, co przedstawiono na wykresie rys. 25. Taka różnica czasowa mogła wynikać z nakładu pracy narzuconego na system OCR, który zmuszony był do zaalokowania pamięci i zainicjowania nowych obiektów dla każdego pobranego rekordu, mimo że przy zastosowanym stronnicowaniu nigdy nie zostały one wykorzystane. Na podstawie powyższych testów można wywnioskować, że mechanizm przedwczesnego załodowania wyników pośrednich nie jest warty użycia w przypadku zastosowania stronnicowania, które dla większej ilości rekordów pozwala osiągnąć lepszy stopień optymalizacji.



Rys. 24 Czas pracy bazy danych w zależności od zastosowania mechanizmu przedwczesnego zaciągania wyników powiązanych przy użytym stronnicowaniu.



Rys. 25 Czas ładownia strony w zależności od zastosowania mechanizmu przedwczesnego zaciągania wyników powiązanych przy użytym stronnicowaniu.

<<jeszcze o rozdzielczości GEOJSON>>

## Testy akceptacyjne

Testy wydajnościowe pozwoliły na ocenę systemu pod kątem technicznym i optymalizacyjnym, jednak nie było to wystarczające do ostateczngo zatwierdzenia zfinalizowania projektu. Wynikało to z wymagań biznesowych narzuconych na aplikację przez klienta, a konkretnie wymaganie o prostocie interfejsu oraz wyspecjalizowanej naturze biznesowej. Pojęcie prostoty interfejsu jest stwierdzeniem bardzo subiektywnym o którym wypowiedzieć mogą się jedynie końcowi użytkownicy. Jednocześnie medyczna natura biznesowa utrudniała autorowi systemu określenie stopnia finalizacji i poprawności systemu i wymagane do tego było zdanie specjalisty. Nie oznacza to, że praca włożna w aspekty techniczne była na darmo, gdyż klient nie przyjąłby systemu, który nie byłby skalowalny i który wykazywałby się dużym czasem przetwarzania instrukcji od użytkownika. Są to jednak cechy, o których klient nietechniczny nie myśli, dopóki takie problemy się nie uwidaczniają.

Jako że w przypadku tego projektu klienci i użytkownicy końcowi byli tą samą grupą, obydwie oceny można było uzyskać w trakcie tych samych konsultacji. W okresie wytwarzania oprogramowania odbyły się trzy spotkania z pracownikami Uniwersyteckiego Centrum Klinicznego, w trakcie których zaprezentowany został im cały system z dokładnym omówieniem wszystkich funkcjonalności przez niego zapewnianych. W ich trakcie lekarze oceniali poszczególne elementy aplikacji, mogli sami użyć przedstawionego prototypu i określali które elementy zostają zaakceptowane, a które należy poddać modefikacjom. Pewną przeszkodą był niski stopień wiedzy technicznej klientów oraz niski stopień wiedzy medycznej autora, jednak udało się dojść do wspólnego języka, który pozwolił na zrozumienie się tych dwóch grup. Taki tryb pracy przekładał się na wymaganie dotyczące otwartości na modyfikacje i elastyczny tryb wytwarzania oprogramowania na każdym poziomie architektoniczym systemu. Cały proces akceptacyjny był utrudniony przez czas w którym projekt był realizowany, zważając na panującą w tym okresie pandemię wirusa COVID-19, który komplikował organizację wspomnianych spotkań z powodów bezpieczeństwa, jak i dużego nakładu pracy narzuconego na przedstawicieli UCK, którzy z natury swojego zawodu byli wyjątkowo zapracowani w bierzącym okresie. Z tego powodu udało się zorganizować tylko jedno spotkanie, w którym wszyscy uczestnicy mogli spotkać się fizycznie na wydziale ETI Politechniki Gdańskiej. Pozostałe dwe spotkanie zostały zorganizowane w sposób online przy wykorzystaniu platformy wideokonferencyjnej Microsoft Teams. Na szczęście dzięki opcjom współdzielania ekranu oraz pewnego stopnia przyzwyczajenia uczestników do kożystania z takiego oprogramowania w trakcie pandemii jakość takich spotkań nie była rażąco pogorszona. Ostatecznie cały system został zaakceptowany przez klienta pod kątem biznesowym i projektu interfejsu oraz wszystkie narzucone wymagania zostały uznane ze zrealizowane.

# Przygotowanie do wdrożenia ???

# Podsumowanie

# WYKAZ LITERATURY

1. Centers for Disease Control and Prevention (CDC) [online]. [dostęp: 07.04.2021]. Dostępny w internecie: <https://www.cdc.gov/>
2. CDC - United States Cancer Statistics: Data Visualizations [online]. [dostęp: 07.04.2021]. Dostępny w internecie: <https://gis.cdc.gov/Cancer/USCS/DataViz.html>
3. CDC – NCHHSTP AtlasPlus [online]. [dostęp: 07.04.2021]. Dostępny w internecie: <https://www.cdc.gov/nchhstp/atlas/index.htm>
4. Strona wydziału CSSE na Johns Hopkins University, USA [online]. [dostęp: 08.04.2011]. Dostępny w internecie: <https://systems.jhu.edu/>
5. COVID-19 Dashboard by the Center for Systems Science and Engineering (CSSE) at Johns Hopkins University (JHU) [online]. [dostęp: 08.04.2021]. Dostępny w internecie: <https://gisanddata.maps.arcgis.com/apps/opsdashboard/index.html#/bda7594740fd40299423467b48e9ecf6>
6. COVID-19 Data Repository by the Center for Systems Science and Engineering (CSSE) at Johns Hopkins University [online]. [dostęp: 08.04.2021]. Dostępny w internecie: <https://github.com/CSSEGISandData/COVID-19>
7. Światowa Organizacja Zdrowia [online]. [dostęp: 08.04.2021]. Dostępny w internecie: <https://www.who.int/>
8. WHO - Coronavirus disease (COVID-19) Situation dashboard [online]. [dostęp: 08.04.2021]. Dostępny w internecie: <https://covid19.who.int/>
9. WHO - World malaria report 2017 - regional profiles [online]. [dostęp: 08.04.2021]. Dostępny w internecie: <https://www.who.int/malaria/publications/world-malaria-report-2017/wmr2017-regional-profiles.pdf>
10. Rinu Gour, „Working Structure of Django MTV Architecture” [online]. [dostęp: 16.06.2021]. Dostępny w internecie: <https://towardsdatascience.com/working-structure-of-django-mtv-architecture-a741c8c64082>
11. Dokumentacja Django – FAQ: General [online]. [dostęp: 16.06.2021]. Dostępny w internecie: <https://docs.djangoproject.com/en/dev/faq/general/#django-appears-to-be-a-mvc-framework-but-you-call-the-controller-the-view-and-the-view-the-template-how-come-you-don-t-use-the-standard-names>
12. Strona główna PostgreSQL - What is PostgreSQL? [online]. [dostęp: 18.06.2021]. Dostępny w internecie: <https://www.postgresql.org/about/>
13. Dokumentacja Django – Strona startowa [online]. [dostęp: 04.08.2021]. Dostępny w internecie: <https://docs.djangoproject.com/en/3.2/>
14. Wykorzystana mapa GEOJSON [online]. [dostęp: 22.02.2021]. Dostępny w internecie: <https://geojson-maps.ash.ms/>
15. Strona główna frameworka stylistycznego Boostrap [online]: [dostęp: 11.08.2021]. Dostępny w internecie: <https://getbootstrap.com/>
16. Strona główna Select2 [online]: [dostęp: 11.08.2021]. Dostępny w internecie: <https://select2.org/>

# WYKAZ RYSUNKÓW

# WYKAZ TABEL

**Dodatek B: Płyta z oprogramowaniem**