**UNIWERSYTET WARMIŃSKO MAZURSKI W OLSZTYNIE**

**WYDZIAŁ MATEMATYKI I INFORMATYKI**

**Adam Trentowski**

**Kierunek: Informatyka**

**Aplikacja do zarządzania gospodarstwem rolnym „FarmGuider”**

**Praca inżynierska**

**wykonana w Katedrze Metod Matematycznych Informatyki**

**pod kierunkiem dr. Pawła Drozdy**

**Olsztyn 2024**

**UNIVERSITY OF WARMIA AND MAZURY IN OLSZTYN**

**FACULTY OF MATHEMATICS AND COMPUTER SCIENCE**

**Adam Trentowski**

**Major: Computer Science**

**Farm management application „FarmGuider”**

**Engineering Thesis**

**written in the Department of Mathematical Methods of Computer Science**

**under supervision of dr Paweł Drozda**

**Olsztyn 2024**

Spis treści

[Streszczenie 4](#_Toc156031631)

[Abstract 4](#_Toc156031632)

[Rozdział I. Wstęp 5](#_Toc156031633)

[1.1 Wprowadzenie 5](#_Toc156031634)

[1.2 Cel i zakres pracy 6](#_Toc156031635)

[1.3 Wykorzystane technologie 6](#_Toc156031636)

[1.3.1 Technologie backendowe 7](#_Toc156031637)

[1.3.2 Technologie frontendowe 8](#_Toc156031638)

[Rozdział II. Specyfikacja systemu 10](#_Toc156031639)

[2.1 Opis systemu 10](#_Toc156031640)

[2.2 Funkcjonalności systemu 12](#_Toc156031641)

[2.2.1 Funkcjonalności ogólne 12](#_Toc156031642)

[2.2.2 Funkcjonalności modułu hodowlanego 13](#_Toc156031643)

[2.2.3 Funkcjonalności modułu uprawowego 14](#_Toc156031644)

[Rozdział III. Diagram przypadków użycia 16](#_Toc156031645)

[3.1 Aktorzy systemowi 16](#_Toc156031646)

[3.2 Diagram przypadków użycia 16](#_Toc156031647)

[3.3 Lista przypadków użycia 19](#_Toc156031648)

[Rozdział IV. Schemat bazy danych 20](#_Toc156031649)

[4.1 Wprowadzenie i schemat 20](#_Toc156031650)

[4.2 Tabele użytkownika i strategia identyfikacji rekordów 21](#_Toc156031651)

[4.3 Tabele modułu uprawowego 22](#_Toc156031652)

[4.4 Tabele modułu hodowlanego 24](#_Toc156031653)

[Rozdział V. Przewodnik użytkownika po aplikacji 27](#_Toc156031654)

# Streszczenie

Niniejsza praca dyplomowa przedstawia projekt i funkcjonalności aplikacji webowej do zarządzania gospodarstwem rolnym „FarmGuider”, której głównym celem jest ułatwienie podejmowania decyzji dotyczących optymalizacji pracy gospodarstwa.

„FarmGuider” oferuje dwa główne moduły. Pierwszy z nich, przeznaczony   
do monitorowania danych o hodowli krów, dostarcza szczegółowych informacji   
o zwierzętach, takich jak informacje o wydajności mlecznej czy przyrostach masy ciała. Funkcjonalności zawarte w tym module, umożliwiają rejestrowanie i śledzenie kluczowych danych, co przekłada się na lepsze zarządzanie hodowlą, a także umożliwia generowanie raportów decyzyjnych.

Drugi moduł aplikacji skupia się na wsparciu osób zarządzających uprawą roślin. Pozwala on, na gromadzenie informacji o polach, zasiewach, zbiorach, stosowanych nawozach, środkach ochrony roślin oraz innych zabiegach. Ta część aplikacji również oferuje generowanie wykresów analizujących efektywność upraw, co pozwala na identyfikację obszarów w gospodarstwie, które wymagają optymalizacji. Dodatkowo aplikacja oferuje kalendarz upraw, który stanowi uzupełnienie wiedzy na temat optymalnych okresów siewnych i żniwnych, upraw dostępnych w systemie.

# Abstract

This diploma thesis presents the design and functionalities of the web application   
for farm management "FarmGuider", whose main goal is to facilitate decision-making regarding the optimization of farm work.

"FarmGuider" offers two main modules. The first one, dedicated to monitoring cow breeding data, provides detailed information about the animals, such as milk yield and weight gain. The functionalities included in this module allow for the registration and tracking of key data, which translates into better herd management and also enables the generation   
of decision-making reports.

The second module of the application focuses on supporting those managing crop cultivation. It allows for the collection of information about fields, sowing, harvesting, applied fertilizers, plant protection products, and other treatments. This part of the application also offers the generation of charts analyzing the efficiency of crops, which helps to identify areas in the farm that require optimization. Additionally, the application offers a crop calendar, which complements the knowledge about optimal sowing and harvesting periods, crops available in the system.

# Rozdział I. Wstęp

## 1.1 Wprowadzenie

W obliczu szybkich przemian zachodzących dzisiejszym świecie, rolnictwo, podobnie jak każda inna branża, stoi przed wyzwaniem zwiększania efektywności i maksymalizacji zysków. Współczesne rolnictwo, wymaga narzędzi, które ułatwiałyby małym i średnim gospodarstwom rolnym nie tylko utrzymanie się na rynku, ale również efektywne konkurowanie z dużymi, bardziej zautomatyzowanymi przedsiębiorstwami rolnymi.

Aplikacja "FarmGuider" została stworzona z myślą o osobach pracujących właśnie   
w takich gospodarstwach, aby zapewnić im dostęp do technologii, które ułatwią optymalizację ich pracy, zwiększając tym samym wydajność i rentowność gospodarstw.   
W erze, gdzie zaawansowane analizy danych i automatyzacja stają się kluczowymi czynnikami sukcesu, "FarmGuider" oferuje rolnikom narzędzia do łatwiejszego zarządzania zasobami, monitorowania hodowli i upraw, a także efektywnego planowania prac związanych z cyklem rolnym.

Dynamika rozwoju rolnictwa, charakteryzująca się z jednej strony rosnącym zapotrzebowaniem na produkty rolne, a z drugiej - zwiększającą się konkurencją i presją kosztową, wymaga od rolników nie tylko dostosowywania się do zmieniających   
się warunków, ale również poszukiwania nowych metod efektywnego zarządzania gospodarstwem. "FarmGuider" odpowiada na te wyzwania, oferując platformę, która integruje technologie zarządzania danymi z praktycznymi aspektami codziennej pracy   
w gospodarstwie. Dzięki temu małe i średnie gospodarstwa uzyskują narzędzie, które   
nie tylko umożliwia im lepsze zarządzanie bieżącymi obowiązkami, ale również ułatwia planowanie przyszłych zadań, oparte na danych i analizach. W ten sposób "FarmGuider"   
nie tylko zwiększa konkurencyjność małych i średnich gospodarstw, ale również przyczynia się do ich stabilności i rozwoju w coraz bardziej zglobalizowanym świecie rolnictwa.

W ramach pierwszego modułu aplikacji, "FarmGuider" koncentruje się na ciągłym monitorowaniu hodowli krów. Moduł ten jest zaprojektowany tak, aby dostarczać rolnikom szczegółowych informacji dotyczących okresów laktacyjnych, wydajności mlecznej oraz zmian masy ciała na przestrzeni życia krowy. Zastosowanie przetwarzania i odpowiedniej prezentacji danych w tym module, pozwala rolnikom na precyzyjne śledzenie wskaźników produktywności, co jest niezbędne do oceny, czy krowy w okresie laktacyjnym jak i bydło mięsne, osiągają oczekiwane wyniki.

Drugi moduł obejmuje wsparcie w zarządzaniu uprawami rolnymi. Umożliwia gromadzenie danych o polach, zasiewach, zbiorach i stosowanych zabiegach, co stanowi fundament dla efektywnego planowania i zarządzania uprawami. Dzięki aplikacji możliwa jest analiza wydajności upraw, co realizowane jest poprzez generowanie raportów.   
Te narzędzia ułatwiają identyfikację obszarów, które wymagają optymalizacji. Ponadto, moduł zawiera kalendarz upraw, który jest pomocnym elementem w planowaniu prac sezonowych.

"FarmGuider" jest odpowiedzią na współczesne wyzwania rolnicze, jako narzędzie   
dla małych i średnich gospodarstw rolnych, dążących do efektywności i konkurencyjności   
w szybko zmieniającym się świecie rolnictwa. Z jego pomocą, zarówno w zakresie hodowli krów, jak i zarządzania uprawami, rolnicy otrzymują dostęp do kluczowych informacji   
i narzędzi analitycznych, które ułatwiają monitorowanie, planowanie i optymalizację pracy.

## 1.2 Cel i zakres pracy

W ramach pracy dyplomowej podejmuję się zadania szczegółowego przedstawienia  
i zilustrowania kolejnych etapów projektowania i implementacji aplikacji webowej „FarmGuider”. Niniejszy rozdział stanowi wstęp do szczegółowej analizy konstrukcji owej aplikacji. Rozpoczynając od jej opisu analitycznego, skupię się na kluczowych aspektach projektowania systemu, w tym na niezbędnych funkcjonalnościach, które aplikacja powinna oferować, oraz na sposobie, w jaki te funkcjonalności wpłyną na codzienne funkcjonowanie gospodarstwa rolnego.

Zasadniczą częścią jest opis analityczny i specyfikacja systemu, w tym analiza wymaganych funkcjonalności aplikacji, przedstawienie diagramu przypadków użycia  
i scenariuszy oraz definicja aktorów systemu. To umożliwia głębsze zrozumienie procesów i mechanizmów działania aplikacji oraz jej potencjalnego wpływu na gospodarstwa rolne.

Następnie, przechodząc do projektu bazy danych, przedstawię sposób, w jaki struktura danych została zaprojektowana i zaimplementowana, aby wspierać funkcjonalność i wydajność aplikacji. Omówię projekt tabel i relacji. Ten etap jest kluczowy dla zrozumienia, w jaki sposób aplikacja przechowuje i zarządza danymi, będąc fundamentem dalszej pracy nad implementacją systemu.

Szczegółowy przewodnik użytkownika, będący końcową częścią pracy, nie tylko instruuje, jak korzystać z aplikacji, ale także jak maksymalizować korzyści płynące z jej używania. Ten element stanowi praktyczne dopełnienie teoretycznej analizy, oferując użytkownikom bezpośredni wgląd w aplikację i jej funkcjonalności.

## 1.3 Wykorzystane technologie

W procesie tworzenia aplikacji webowej "FarmGuider", ważnym aspektem był wybór odpowiednich technologii. Technologie te zostały podzielone na dwie kategorie, odpowiadające za różne aspekty aplikacji: Backend i Frontend. Każda z kategorii odgrywa inną rolę w budowie i funkcjonowaniu aplikacji.

### 1.3.1 Technologie backendowe

* **Java 20**: jest silnie typowanym językiem programowania ogólnego zastosowania. Język ten, koncentruje się na obiektowości i przenośności kodu pomiędzy różnymi platformami. W wersji 20 posiada już szereg funkcjonalności, ułatwiających i przyspieszających pisanie zrozumiałego i wydajnego kodu, takich jak wyrażenia lambda czy strumienie. W aplikacji „FarmGuider”, Java została przeze mnie użyta do stworzenia backendu, właśnie ze względu na jej zalety oraz doświadczenie, jakie posiadam przy pracy z tym językiem.
* **Gradle 8.2.1**: to narzędzie, służące do automatyzacji procesów kompilacji, testowania i wdrażania oprogramowania. Jego głównymi atutami są czytelny język konfiguracji oparty na Kotlinie, oraz szybkość działania. Plik konfiguracyjny, korzystający z Kotlina, ułatwia zarządzanie zależnościami czyniąc ten proces bardziej intuicyjnym niż w przypadku tradycyjnych narzędzi takich jak Maven. Wybór Gradle'a był również podyktowany wyższą prędkością działania w porównaniu z Maven’em.
* **SpringBoot 3.1.2**: jest to zaawansowany framework, który zapewnia wszechstronne wsparcie dla tworzenia aplikacji, takie jak zarządzanie transakcjami czy wsparcie REST API do komunikacji z frontendem. Zawiera ponadto wbudowane serwery aplikacyjne, co eliminuje potrzebę zewnętrznej konfiguracji serwera. Zapewnia również wsparcie w zakresie bezpieczeństwa, udostępniając bibliotekę **SpringSecurity 6.2**, która posiada szerokie możliwości konfiguracji autentykacji i autoryzacji użytkowników, zabezpieczając aplikację przed nieautoryzowanym dostępem i atakami.
* **PostgreSQL 15**: to open source'owy, relacyjny system baz danych, który wykorzystuje i rozszerza język SQL. Ceniony jest za swoją niezawodność, wydajność i wszechstronność, dzięki czemu jest idealnym wyborem dla aplikacji, które wymagają stabilności. W wersji 15, PostgreSQL oferuje jeszcze bogatszy zestaw narzędzi i ulepszeń, które zapewniają optymalizację wydajności, w tym lepsze zarządzanie pamięcią i szybsze przetwarzanie zapytań.
* **Flyway 9.20.1**: jest narzędziem służącym do zarządzania migracjami baz danych. Umożliwia ono deweloperom wersjonowanie schematu bazy danych, co jest istotne w kwestii śledzenia i zarządzania zmianami w strukturze danych. Poprzez zastosowanie kontrolowanych skryptów migracyjnych, Flyway gwarantuje, że każda zmiana – od prostego dodania kolumny po skomplikowane modyfikacje schematu – jest rejestrowana, możliwa do śledzenia oraz odwracalna.
* **Open API/Swagger**: to standard używany do opisywania i wizualizowania usług webowych. Z pomocą Open API, można tworzyć przystępną i czytelną dokumentację dla API budowanej aplikacji. Co ważne, dokumentacja wytworzona z pomocą Open API, generowana jest automatycznie, na podstawie kodu.
* **JWT (JSON Web Token)**: to kompaktowy, bezpieczny sposób przekazywania informacji między stronami jako obiekt JSON. Struktura JWT składa się (zazwyczaj) z trzech części: nagłówka (header), ładunku (payload), zawierającego istotne informacje, oraz sygnatury (signature), która zapewnia integralność danych i weryfikację autentyczności tokena. Sygnatura generowana jest na podstawie zawartości tokena oraz przy użyciu tajnego klucza. To zapewnia, że tokeny są nie tylko prawdziwe, ale i chronione przed modyfikacją przez nieautoryzowane strony. W aplikacji „FarmGuider”, przy pomocy JWT, utworzony został mechanizm autoryzacji i uwierzytelniania, zapewniający bezpieczną transmisję danych między klientem a serwerem.

### 1.3.2 Technologie frontendowe

* **NPM 9.6.7**: czyli Node Package Manager, jest popularnym i szeroko stosowanym systemem zarządzania pakietami dla środowiska JavaScript. Jest to istotne narzędzie w środowisku frontendowym, umożliwiające deweloperom efektywne zarządzanie bibliotekami i zależnościami w projektach. NPM ułatwia instalację, aktualizację, konfigurację oraz usuwanie pakietów JavaScript, co jest niezbędne dla utrzymania efektywności i porządku w kodzie aplikacji.
* **React 18.2.0**: będący jedną z najpopularniejszych bibliotek JavaScript, został użyty w aplikacji „FarmGuider” do tworzenia kompaktowych i reaktywnych interfejsów użytkownika. Jego podejście oparte na komponentach umożliwia modularną i łatwą w utrzymaniu strukturę kodu, co znacznie upraszcza proces tworzenia skomplikowanych interfejsów. Dodatkowo, w połączeniu z narzędziem do budowania aplikacji, **Vite 4.4.5**, React oferuje jeszcze szybszy proces rozwoju dzięki błyskawicznemu odświeżaniu modułów i optymalizacji kompilacji.
* **Typescript 5.0.2**: będący nadzbiorem języka JavaScript, jest używany w aplikacji „FarmGuider” jako główny język programowania dla frontendu. Jego kluczową cechą jest silne typowanie, które znacznie poprawia czytelność kodu, ułatwia wykrywanie błędów na wczesnym etapie rozwoju i zwiększa ogólną niezawodność aplikacji. Wybór Typescript'a w miejsce czystego JavaScript’u został podyktowany potrzebą łatwiejszego zarządzania kodem.
* **HTML** i **CSS**: są podstawowymi technologiami wykorzystywanymi w aplikacji „FarmGuider” do budowania interfejsów użytkownika. HTML jest używany do strukturyzowania treści w aplikacji webowej, tworząc szkielet każdej strony, wliczając w to teksty, obrazy i inne elementy multimedialne. Jest to język znaczników, który określa, jak poszczególne elementy strony są wyświetlane w przeglądarce internetowej. Z kolei CSS jest używany do stylizowania tych elementów, umożliwiając tworzenie estetycznych i responsywnych interfejsów użytkownika. Dzięki CSS możliwe jest definiowanie wyglądu stron, w tym kolorów, czcionek i układów itd.  
    
  Oprócz wyżej wymienionych technologii, do stworzenia responsywnej, pozwalającej na wybór języka aplikacji, posłużyły również biblioteki uzupełniające bibliotekę React:
* **i18n**: jest to narzędzie służące do internacjonalizacji, które pozwala na przechowywanie tekstów interfejsu w różnych językach w uporządkowany i łatwo dostępny sposób. Dzięki i18n, „FarmGuider” może dynamicznie zmieniać język interfejsu użytkownika, dostosowując się do preferencji i potrzeb użytkowników. Użycie i18n nie tylko zwiększa dostępność aplikacji dla szerszej publiczności, ale także ułatwia zarządzanie treścią i aktualizacje. Zmiany lub dodatki do tekstu interfejsu można łatwo wprowadzać w jednym miejscu, a biblioteka automatycznie propaguje te zmiany we wszystkich obsługiwanych językach.
* **MUI (Material-UI)**: to wszechstronna biblioteka komponentów UI stworzona dla React, używana w aplikacji „FarmGuider” do budowania estetycznych i funkcjonalnych interfejsów użytkownika. Bazując na zasadach projektowania Material Design, MUI oferuje szeroki zakres gotowych do użycia komponentów, takich jak przyciski, formularze, karty i inne elementy interaktywne, które można łatwo dostosować do potrzeb projektu. Dzięki tej bibliotece deweloperzy mogą szybko tworzyć spójne i atrakcyjne interfejsy, nie tracąc czasu na projektowanie i implementację podstawowych elementów od zera.
* **Recharts**: jest biblioteką wykresów stworzoną specjalnie dla React, która została wykorzystana w aplikacji „FarmGuider” do wizualizacji raportów. Jest to narzędzie oparte na komponentach React, co ułatwia integrację wykresów z resztą aplikacji i zapewnia spójność interfejsu użytkownika. Recharts oferuje szeroki wachlarz typów wykresów, w tym wykresy słupkowe, liniowe, obszarowe, kołowe i inne, co pozwala na elastyczne i atrakcyjne przedstawianie danych.
* **React-Parallax**: biblioteka, która została użyta w aplikacji „FarmGuider” do wzbogacenia interfejsu użytkownika o efekt paralaksy, podnosząc jego estetykę. Efekt paralaksy to technika w projektowaniu stron internetowych, gdzie tło porusza się wolniej niż elementy pierwszoplanowe podczas przewijania strony, tworząc złudzenie głębi i dodając wrażenia trójwymiarowości dla użytkownika. Biblioteka React-Parallax umożliwia łatwą implementację tego efektu w aplikacjach React.
* **Notistack**: jest biblioteką używaną w aplikacji „FarmGuider” do efektywnego zarządzania powiadomieniami typu Snackbar w środowisku React. Snackbar’y to krótkie komunikaty, które pojawiają się na ekranie, aby przekazać użytkownikom istotne informacje, takie jak potwierdzenia akcji, ostrzeżenia lub błędy, w sposób subtelny i nieinwazyjny. Są one często wykorzystywane w interfejsach użytkownika do zapewnienia szybkiej i zrozumiałej komunikacji bez przerywania przepływu pracy użytkownika.

# Rozdział II. Specyfikacja systemu

## 2.1 Opis systemu

Podczas prac nad aplikacją „FarmGuider”, główne założenia skupiały się na usprawnieniu procesów decyzyjnych w gospodarstwie rolnym oraz na zapewnieniu wysokiej wygody użytkowania. Fundamentalnym aspektem projektu było również zagwarantowanie bezpieczeństwa systemu, co ma kluczowe znaczenie w ochronie danych i zapobieganiu dostępu nieautoryzowanych osób do zasobów i funkcji aplikacji.

Z myślą o zapewnieniu bezpiecznego dostępu dla użytkowników, zanim użytkownik uzyska możliwość korzystania z pełnej funkcjonalności aplikacji, wymagane jest jego zarejestrowanie oraz zalogowanie. Proces ten tworzy bezpieczną sesję, co stanowi istotny element ochrony danych użytkownika i zapobiega nieautoryzowanemu dostępowi.

Po pomyślnym zalogowaniu się, użytkownik jest przenoszony na stronę główną aplikacji, która służy jako szybkie podsumowanie informacji o gospodarstwie. Strona ta zawiera dane na temat hodowli, takie jak liczebność zwierząt w gospodarstwie, aktualne informacje o udojach oraz przypomnienia dotyczące porannego i wieczornego udoju. Ponadto, użytkownik ma dostęp do podsumowania dotyczącego pól uprawnych.

Przechodząc dalej, prace gospodarcze zostały skategoryzowane jako dwa osobne moduły odpowiedzialne za: hodowlę bydła i uprawy roślinne. Oba te moduły funkcjonują jako niezależne kategorie, dostosowane do specyficznych potrzeb i charakterystyk każdej z tych dziedzin. Dzięki temu, użytkownik może efektywnie zarządzać różnorodnymi aspektami gospodarstwa, korzystając z dedykowanych narzędzi i funkcji zaprojektowanych specjalnie dla hodowli oraz upraw.

Pierwszym elementem aplikacji „FarmGuider” jest moduł hodowli, który został zaprojektowany z myślą o zapewnieniu efektywnego zarządzania hodowlą bydła. Aby skorzystać z funkcjonalności tego modułu, użytkownik musi najpierw utworzyć stado. System pozwala na utworzenie wielu stad, co umożliwia skategoryzowanie i lepszą organizację zwierząt hodowlanych. Po utworzeniu stada, użytkownik ma dostęp do interfejsu z wyszukiwarką krów, która służy jako główny punkt zarządzania hodowlą.

W tym widoku użytkownik ma możliwość edytowania aktualnie wybranego stada, dodawania nowych i usuwania niepotrzebnych stad, może również wybrać konkretne stado z listy, aby do niego przejść. W miejscu tym znajduje się również przycisk dodawania nowej krowy do stada. Po dodaniu krowy do systemu, staje się ona widoczna w wyszukiwarce – chyba że ustawione filtry wyszukiwania nie obejmują parametrów nowego zwierzęcia.

Wyszukiwarka umożliwia sortowanie i wyszukiwanie krów według różnych podstawowych kryteriów, ale też według takich danych jak aktualna waga czy ilość mleka z ostatniego udoju, które znalazły się tu dla wygody użytkownika. Ponadto, aby uprościć korzystanie z aplikacji, każdy wpis krowy w wyszukiwarce posiada cztery przyciski funkcyjne, które pozwalają na: dodanie udoju, wprowadzenie pomiaru wagi, usunięcie krowy ze stada, a także przejście do bardziej szczegółowego widoku krowy.

Widok szczegółowych informacji o krowie składa się z pięciu zintegrowanych sekcji, które wspólnie oferują kompleksowy zestaw funkcji zarządzania zwierzęciem. Pierwsza sekcja zawiera podstawowe informacje o krowie, z możliwością ich edycji. Następna sekcja przedstawia historię udojów krowy, prezentowaną w formie przystępnej listy. Każdy zapis udoju w tej sekcji może być edytowany lub usunięty. Umieszczony jest tu również przycisk do dodawania nowych udojów. W przypadku byków, funkcje związane z udojami są zablokowane. Kolejna sekcja to lista pomiarów wagi, która, podobnie jak lista udojów, pozwala na edycję i usuwanie wprowadzonych danych. Lista ta również zawiera funkcję dodawania nowych pomiarów wagi. Czwarty element to sekcja raportu wydajności mlecznej, zawierająca krzywą laktacji krowy. Wykres ten generowany jest na podstawie wprowadzonych okresów laktacji, co pozwala na analizę wydajności mlecznej. Użytkownik ma tutaj możliwość dodawania, edytowania i usuwania okresów laktacyjnych oraz wybierania konkretnych okresów do szczegółowej analizy. Ostatnią sekcją jest wykres zmian masy ciała zwierzęcia, który pozwala na wizualne śledzenie dynamiki przyrostu masy, identyfikując ewentualne zatrzymania wzrostu lub spadki. Interfejs szczegółów krowy zawiera także przycisk, który umożliwia usunięcie zwierzęcia z systemu.

Moduł dedykowany uprawom roślin, który stanowi element efektywnego zarządzania gospodarstwem, został stworzony z myślą o zapewnieniu narzędzi do planowania i zarządzania uprawami w gospodarstwie. Podobnie jak w przypadku modułu hodowli bydła, pierwszym krokiem w korzystaniu z tego modułu jest dodanie przynajmniej jednego pola uprawnego.

Po dodaniu pola, użytkownik otrzymuje dostęp do wyszukiwarki pól, która zawiera również przycisk umożliwiający dodanie kolejnych pól. Każdy wpis w wyszukiwarce zawiera krótkie podsumowanie informacji o polu, w tym jego nazwę dla łatwiejszej identyfikacji, powierzchnię oraz opcjonalnie klasę gruntu. Ponadto, każdy wpis stanowi bezpośrednie przekierowanie do szczegółowego widoku danego pola.

Widok szczegółowych informacji o polu, składa się z czterech sekcji, które współdziałają w celu usprawnienia gromadzenia i analizy danych o uprawach oraz stosowanych zabiegach. Pierwsza sekcja koncentruje się na informacjach podstawowych o polu. Użytkownik ma możliwość edycji tych informacji oraz usuwania pola z systemu. Druga sekcja prezentuje listę historii upraw prowadzonych na danym polu. Każdy wpis na liście upraw zawiera podstawowe dane o uprawie, a także przewidywany okres zbiorów oraz szacowany plon, obliczony na podstawie danych systemowych. Dla każdej uprawy dostępne są przyciski umożliwiające dodanie informacji o zbiorach (co jest równoznaczne z zakończeniem uprawy) oraz usunięcie uprawy z listy. Lista wyposażona jest w przycisk do dodawania nowych upraw. Trzecia część interfejsu to lista zabiegów stosowanych dla wybranej uprawy. Po zaznaczeniu konkretnej uprawy w sekcji z listą upraw, użytkownik otrzymuje dostęp do szczegółowych informacji o stosowanych nawozach, opryskach i innych zabiegach. Każdy wpis zawiera podstawowe informacje o zabiegu, uzupełnione o opcjonalne informacje dodatkowe. Każdy wpis posiada przyciski do edycji oraz usuwania zabiegu. Dostępny jest również przycisk umożliwiający dodanie nowego zabiegu do wybranej uprawy, odblokowywany dopiero po wybraniu uprawy. Zamykający całość element to raport efektywności upraw. Prezentuje on wykres słupkowy zestawień przewidywanych plonów w stosunku do plonów rzeczywistych, który pozwala na ocenę skuteczności prowadzonych upraw i planowanie przyszłych działań.

Dodatkowym elementem modułu upraw jest kalendarz upraw, który jest zbiorem wiedzy na temat optymalnych okresów siewów i zbiorów każdej uprawy wprowadzonej do systemu. Kalendarz graficznie przedstawia owe okresy na przestrzeni 12 miesięcy.

„FarmGuider” posiada również widok profilu użytkownika, który umożliwia zarządzanie danymi osobistymi. Dostępne są tu różne opcje edycji danych wprowadzonych podczas procesu rejestracji, z wyjątkiem adresu e-mail, który pozostaje stały. Użytkownik ma możliwość aktualizacji swoich danych osobowych, w tym dodawania i edytowania adresu zamieszkania. Szczególnym przypadkiem edycji danych, jest zmiana hasła, która wymaga podania obecnego hasła oraz wprowadzenia nowego. W profilu znajduje się również przycisk umożliwiający usunięcie konta użytkownika. Proces ten jest również zabezpieczony poprzez wymóg podania aktualnego hasła, aby zapewnić, że decyzja o usunięciu konta jest świadoma i autoryzowana przez użytkownika.

Co do ogólnej konstrukcji aplikacji, „FarmGuider” został zaprojektowana z myślą o osobach pracujących w rolnictwie, które potrzebują łatwego dostępu do systemu zarządzania gospodarstwem w codziennej pracy. Aplikacja jest w pełni responsywna, co oznacza, że jej interfejs dostosowuje się do różnych rozmiarów ekranów, umożliwiając użytkowanie na urządzeniach mobilnych, takich jak smartfony. Taka funkcjonalność jest szczególnie przydatna dla osób, które chcą na bieżąco wprowadzać dane, na przykład dotyczące udojów lub pomiarów wagi bydła.

Dodatkowo, aplikacja została wyposażona w dwie wersje językowe – polską i angielską. Możliwość zmiany języka aplikacji w menu zapewnia większą dostępność i komfort użytkowania dla szerokiego grona odbiorców.

## 2.2 Funkcjonalności systemu

Na podstawie opisu systemu, zostały wylistowane funkcjonalności systemu z podziałem na moduły. Wszystkie funkcjonalności poza rejestracją, logowaniem i zmianą języka interfejsu aplikacji wymagają autoryzacji. Funkcjonalności rozumiane jako *read* z zestawu czterech podstawowych funkcji *CRUD* zostały w większości pominięte.

### 2.2.1 Funkcjonalności ogólne

* Rejestracja użytkownika – utworzenie konta w aplikacji, na podstawie wymaganych danych takich jak: imię, nazwisko, adres e-mail i hasło. Po pomyślnej rejestracji użytkownik jest autentykowany i zostaje utworzona sesja w aplikacji.
* Logowanie użytkownika – funkcja umożliwiająca zarejestrowanym użytkownikom na zautentykowanie się i utworzenie sesji w aplikacji poprzez podanie adresu e-mail i hasła, celem uzyskania dostępu do wszystkich funkcjonalności aplikacji.
* Wylogowanie użytkownika – funkcja pozwalająca na bezpieczne zakończenie sesji w aplikacji.
* Edycja danych użytkownika – umożliwia użytkownikom na zaktualizowanie danych osobowych, a także na wprowadzenie i edycję danych adresowych.
* Zmiana hasła – opcja pozwalająca na zmianę hasła do konta, pod warunkiem wprowadzenia aktualnego hasła i podania nowego.
* Usunięcie konta – funkcja umożliwiająca użytkownikom trwałe usunięcie ich konta z aplikacji wraz ze wszystkim powiązanymi danymi. Proces ten wymaga potwierdzenia poprzez podanie aktualnego hasła.
* Zmiana języka interfejsu aplikacji – funkcja ta pozwala użytkownikom na wybór preferowanego języka interfejsu aplikacji.

### 2.2.2 Funkcjonalności modułu hodowlanego

* Dodanie stada - umożliwia użytkownikom utworzenie nowego stada poprzez podanie nazwy stada.
* Edycja stada – pozwala na zmianę nazwy stada.
* Usunięcie stada – funkcja umożliwiająca trwałe usunięcie stada z aplikacji wraz ze wszystkimi przypisanymi do stada zwierzętami.
* Wybór stada – umożliwia wybór aktualnie przeglądanego stada.
* Dodanie krowy do stada – opcja pozwalająca na dodanie krowy do aktualnie wybranego stada, poprzez podanie wymaganych danych takich jak: numer kolczyka, płeć zwierzęcia i data urodzenia. Opcjonalnie można wprowadzić imię krowy.
* Wyszukanie krów - użytkownicy mogą wyszukiwać krowy w ramach aktualnie wybranego stada na podstawie kryteriów takich jak: numer kolczyka, płeć, imię krowy, data urodzenia krowy od-do, aktualna waga krowy od-do, ilość mleka z ostatniego udoju od-do. Ponadto można zmienić kryterium i kierunek sortowania spośród wcześniej podanych, a także zmienić ilość wyświetlanych na stronę wyników oraz przełączać się pomiędzy samymi stronami z wynikami.
* Edycja danych krowy – pozwala na aktualizację danych zwierzęcia.
* Usunięcie krowy – umożliwia trwałe usunięcie zwierzęcia z aplikacji wraz ze wszystkimi przypisanymi do krowy udojami i pomiarami wagi ciała.
* Dodanie udoju – użytkownicy mogą tworzyć nowy wpis dotyczący udoju aktualnie wybranej krowy, zawierający wymagane informacje takie jak: data udoju wraz z godziną i ilość litrów mleka. Opcjonalnie mogą podać również czas trwania udoju. Opcja ta jest niedostępna jeśli wybrane zwierzę jest samcem.
* Edycja udoju – funkcja umożliwiająca modyfikację wszystkich danych już utworzonego udoju.
* Usunięcie udoju – pozwala na trwałe usunięcie udoju.
* Dodanie pomiaru wagi – umożliwia dodanie nowego wpisu dotyczącego pomiaru wagi aktualnie wybranej krowy, zawierającego wymagane dane dotyczące daty pomiaru oraz masy zwierzęcia.
* Edycja pomiaru wagi – użytkownicy mogą zmienić dane dotyczące istniejącego pomiaru wagi.
* Usunięcie pomiaru wagi – funkcja pozwalająca na trwałe usunięcie pomiaru wagi.
* Dodanie okresu laktacji – pozwala na dodanie okresu laktacji, gdzie wymogiem jest podanie daty początku okresu. Podanie daty końca jest opcjonalne, jeśli nie jest podana, oznacza to, że okres trwa od daty początku do daty obecnej. Innymi słowy okres laktacji ciągle trwa. Opcja ta jest niedostępna jeśli wybrane zwierzę jest samcem.
* Edycja okresu laktacji – funkcja umożliwiająca zaktualizowanie wszystkich danych okresu laktacji. Może być również traktowana jako opcja zakończenia okresu laktacji.
* Usunięcie okresu laktacji – umożliwia trwałe usunięcie okresu laktacji.
* Odczyt wykresu wydajności mlecznej krowy – by użytkownik mógł odczytać raport dotyczący wydajności mlecznej krowy, musi dodać przynajmniej jeden okres laktacji. Jeśli jest dodany jeden, nie będzie możliwości wyboru innego okresu, a wykres będzie zawsze dotyczył tylko tego jednego okresu. Jeśli jest dodany więcej niż jeden okres, wówczas pojawia się możliwość wyboru okresu laktacyjnego, dla którego wykres ma być generowany. Domyślnie wykres generuje się dla okresu laktacyjnego z najnowszą datą początku. Jeśli w danym okresie laktacyjnym nie ma dodanych żadnych udojów, pojawia się stosowna informacja. Odpowiednia informacja pojawia się również jeśli zwierzę jest samcem.
* Odczyt wykresu przyrostów wagi – użytkownik ma możliwość odczytu raportu zmian masy ciała zwierzęcia, jeśli został dodany przynajmniej jeden pomiar wagi ciała. Wykres dotyczy okresu od daty urodzenia zwierzęcia do daty obecnej.

### 2.2.3 Funkcjonalności modułu uprawowego

* Dodanie pola – możliwość dodania pola poprzez wprowadzenie wymaganych danych: nazwy pola i powierzchni. Podanie klasy gruntu jest opcjonalne.
* Wyszukanie pól – opcja filtrowania dodanych pól na podstawie parametrów: nazwy pola, powierzchni od-do oraz klasy ziemi. Funkcja wyszukiwania pól jest dostępna, gdy użytkownik doda przynajmniej jedno pole.
* Edycja pola – użytkownicy mogą zaktualizować dane dotyczące pola.
* Usunięcie pola – funkcja umożliwiająca trwałe usunięcie pola wraz ze wszystkimi przypisanymi do pola uprawami i zabiegami stosowanymi na tych uprawach.
* Dodanie uprawy – pozwala na dodanie uprawy do aktualnie wybranego pola. Warunkiem utworzenia uprawy jest podanie daty siewu oraz wybór typu uprawy z dostępnych w systemie typów upraw.
* Dodanie zbioru uprawy – opcja uzupełnienia wybranej uprawy o informacje: datę zbioru oraz plon. Jeśli użytkownik chce dodać zbiór uprawy – musi podać obie dane.
* Usunięcie uprawy – funkcja dająca możliwość trwałego usunięcia wybranej uprawy i wszystkich związanych z nią zabiegów.
* Odczyt zabiegów uprawy – możliwość odczytu listy zastosowanych na danej uprawie zabiegów. Użytkownik musi najpierw wybrać uprawę z listy upraw. Jeśli dana uprawa nie posiada żadnych zabiegów, zostanie wyświetlona stosowna informacja.
* Dodanie zabiegu – funkcja pozwalająca na dodanie zabiegu do aktualnie wybranej uprawy. Aby dodać zabieg, użytkownik musi podać jego nazwę, datę wykonania, a także ilość. Opcjonalnie użytkownik może wprowadzić informacje dodatkowe.
* Edycja zabiegu – opcja aktualizacji danych zabiegu.
* Usunięcie zabiegu – funkcja umożliwiająca trwałe usunięcie zabiegu.
* Odczyt wykresu efektywności upraw – użytkownik ma możliwość odczytu raportu efektywności upraw, jeżeli została dodana przynajmniej jedna uprawa, do której został dodany zbiór. Jeśli wykres jest generowany, to wyświetla dane dla wszystkich upraw.
* Odczyt kalendarza upraw – możliwość odczytania informacji o optymalnych sezonach siewnych i żniwnych, wszystkich dostępnych w systemie typów upraw.

# Rozdział III. Diagram przypadków użycia

Diagram przypadków użycia, czyli DPU, to schemat przedstawiania funkcjonalności systemu, z perspektywy osób, z niego korzystających. DPU skupia się na sposobach, w jakie użytkownicy (aktorzy) mogą wchodzić w interakcję z aplikacją.

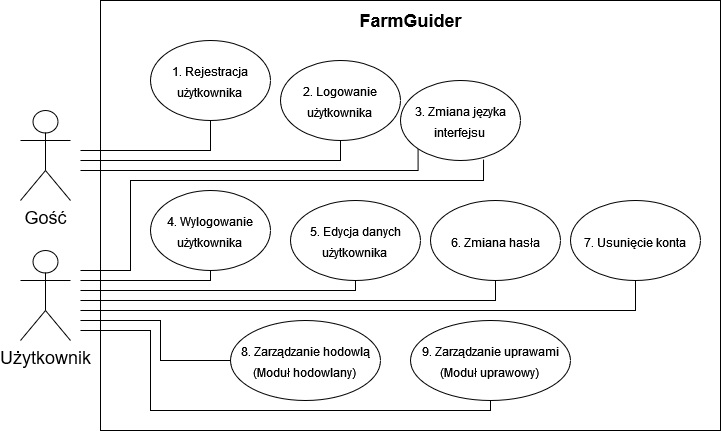
## 3.1 Aktorzy systemowi

W kontekście aplikacji „FarmGuider” można wyróżnić dwóch aktorów:

* Gość (niezalogowany użytkownik systemu) – ogólnie określona osoba, która wchodzi w interakcję z systemem, ale nie jest zalogowana/zarejestrowana. Gość może przeglądać publicznie dostępne informacje, ale nie ma dostępu do funkcjonalności wymagających autoryzacji.
* Użytkownik (zalogowana osoba) – aktor, który posiada konto w systemie i jest zalogowany. Może korzystać z pełnego zakresu funkcjonalności aplikacji.

## 3.2 Diagram przypadków użycia

Dla aplikacji „FarmGuider” w celu uproszczenia zrozumienia, co zachodzi na diagramie, DPU został rozdzielony w sposób podobny, do podziału funkcjonalności systemu na funkcjonalności ogólne i funkcjonalności modułowe.



Powyższy diagram ilustruje rozdział osób korzystających z aplikacji „FarmGuider”,  
na użytkowników zalogowanych i niezalogowanych (gości). Ponadto zostały ukazane dwa szeroko rozumiane PU – przypadki użycia, dotyczące zarządzania hodowlą i zarządzania uprawami, odnoszące się do korzystania z funkcjonalności modułu hodowlanego i funkcjonalności modułu uprawowego.

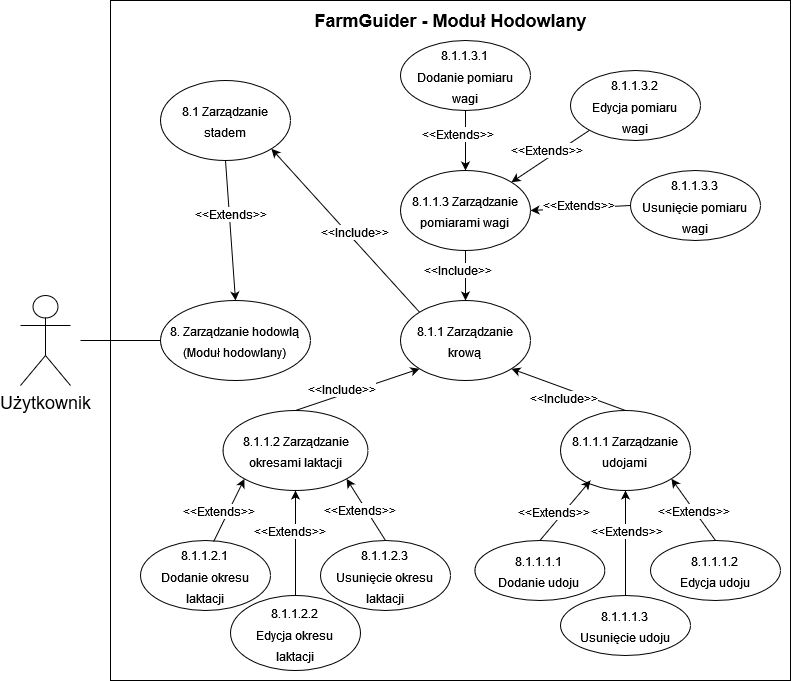


Diagram dotyczący modułu hodowlanego również został dodatkowo podzielony względem przypadków użycia. Po pierwsze, PU zostały pogrupowane na PU odnoszące się do funkcjonalności z tej samej grupy (dotyczące zarządzania tą samą dziedziną funkcjonalną).  
Po drugie, dziedziny dotyczące zarządzania stadem i zarządzania krową zostały również zilustrowane na osobnym diagramie dla łatwiejszego odczytywania diagramów.

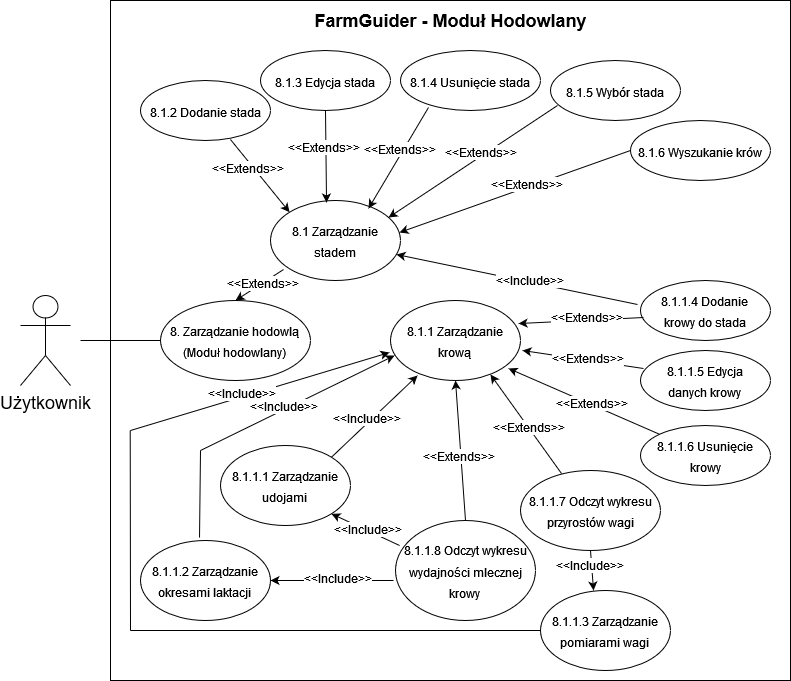


Diagram dotyczący dziedzin zarządzania stadem i zarządzania krową, z racji na niemałą ilość powiązanych ze sobą funkcjonalności, został przedstawiony osobno. Pokazane na nim PU uszczegóławiają poprzednie diagramy, ukazując zależności między funkcjonalnościami.

Przykładem może być odczyt wykresu wydajności mlecznej krowy dotyczący dziedziny zarządzania krową. Zawiera on dziedziny zarządzania udojami oraz zarządzania okresami laktacji, z racji na wymóg dodania co najmniej jednego udoju dla danej krowy oraz utworzenia przynajmniej jednego okresu laktacyjnego.

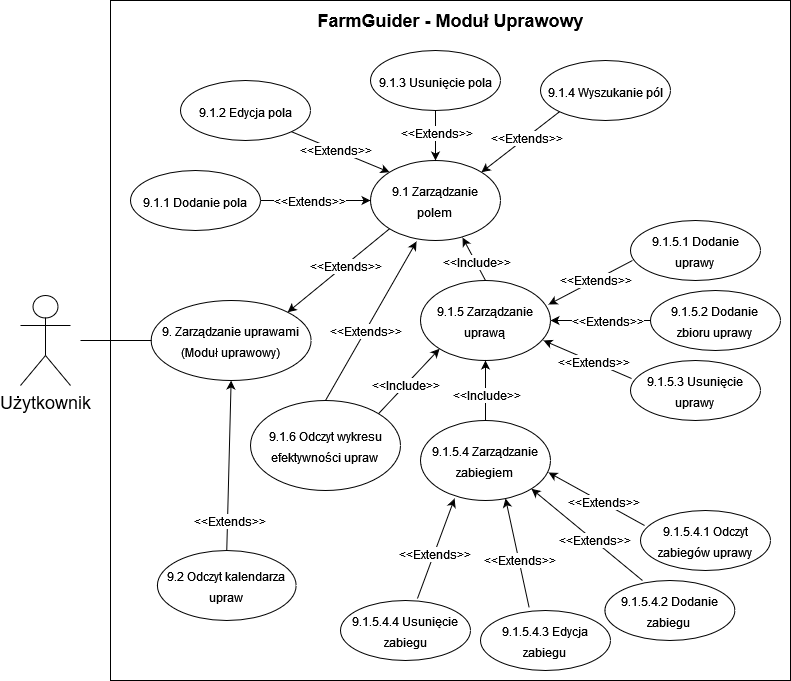


Diagram dotyczący modułu uprawowego, podobnie do diagramu dla modułu hodowlanego, również został podzielony względem dziedzin funkcjonalnych. Poziom skomplikowania jest tu już jednak niższy, dlatego też diagram ten nie został podzielony na bardziej szczegółowe ilustracje.

## 3.3 Lista przypadków użycia

W tym podrozdziale zaprezentowana zostanie szczegółowa analiza różnych scenariuszy, z którymi użytkownicy aplikacji „FarmGuider” mogą się spotkać. Lista przypadków użycia została podzielona na trzy główne kategorie, co odzwierciedla wcześniejszy podział funkcjonalności. Pierwsza część to lista PU ogólnych. Druga część skupia się na przypadkach użycia przeznaczonych dla modułu hodowlanego, odnosząc się do funkcjonalności i procesów związanych z zarządzaniem hodowlą zwierząt. Ostatnia część przedstawia listę PU dla modułu uprawowego, koncentrując się na scenariuszach związanych z zarządzaniem uprawami roślinnymi.

### 3.3.1 Lista PU ogólnych

|  |  |
| --- | --- |
| Przypadek użycia | **1. Rejestracja użytkownika** |
| Aktor | Gość |
| Cel | Rejestracja nowego konta użytkownika w systemie |
| Warunki wstępne | Gość nie jest zalogowany |
| Scenariusz główny | 1) Będąc na stronie głównej, gość naciska przycisk „rozpocznij”  2) System wyświetla formularz logowania i rejestracji  3) Gość wybiera formularz rejestracji  4) Gość wypełnia formularz wymaganymi danymi: imieniem, nazwiskiem, adresem email oraz hasłem  5) Gość naciska przycisk „zarejestruj”  6a) System tworzy konto użytkownika |
| Scenariusz alternatywny | 6b) System informuje gościa o niekompletności lub nieprawidłowości danych i prosi o ich uzupełnienie |
| Po zakończeniu | Gość zostaje zalogowany i przeniesiony na stronę główną po zalogowaniu, gdzie wyświetlany jest komunikat o pomyślnej rejestracji. |

|  |  |
| --- | --- |
| Przypadek użycia | **2. Logowanie użytkownika** |
| Aktor | Gość |
| Cel | Zalogowanie się do systemu |
| Warunki wstępne | - Gość nie jest zalogowany  - Gość posiada konto w systemie |
| Scenariusz główny | 1) Będąc na stronie głównej, gość naciska przycisk „rozpocznij”  2) System wyświetla formularz logowania i rejestracji  3) Gość wybiera formularz logowania  4) Gość wypełnia formularz wymaganymi danymi: adresem email oraz hasłem  5) Gość naciska przycisk „zaloguj”  6a) System loguje gościa |
| Scenariusz alternatywny | 6b) System informuje gościa o niekompletności lub nieprawidłowości danych i prosi o ich uzupełnienie |
| Po zakończeniu | Gość zostaje przeniesiony na stronę główną po zalogowaniu, gdzie wyświetlany jest komunikat o pomyślnym logowaniu. |

|  |  |
| --- | --- |
| Przypadek użycia | **3. Zmiana języka interfejsu** |
| Aktor | Gość/Użytkownik |
| Cel | Zmiana języka interfejsu aplikacji |
| Warunki wstępne | *brak* |
| Scenariusz główny | 1) Gość/użytkownik naciska przycisk „język”, widoczny na pasku nawigacyjnym  2) System wyświetla listę dostępnych języków aplikacji  3) Gość/użytkownik wybiera interesujący go język |
| Scenariusz alternatywny | *brak* |
| Po zakończeniu | System zmienia język interfejsu całej aplikacji na wybrany. |

|  |  |
| --- | --- |
| Przypadek użycia | **4. Wylogowanie użytkownika** |
| Aktor | Użytkownik |
| Cel | Wylogowanie się z systemu |
| Warunki wstępne | Użytkownik jest zalogowany |
| Scenariusz główny | 1a) Użytkownik naciska przycisk „wyloguj”, widoczny na pasku nawigacyjnym  2) System wylogowuje użytkownika z systemu |
| Scenariusz alternatywny | 1a) Użytkownik będąc na stronie głównej, naciska na rozwijane menu  2a) Użytkownik wybiera opcję „wyloguj”  3a) System wylogowuje użytkownika |
| Po zakończeniu | Gość zostaje przeniesiony na stronę główną przed zalogowaniem, gdzie wyświetlany jest komunikat o pomyślnym wylogowaniu. |

|  |  |
| --- | --- |
| Przypadek użycia | **5. Edycja danych użytkownika** |
| Aktor | Użytkownik |
| Cel | Zmiana danych osobowych lub adresowych w systemie |
| Warunki wstępne | Użytkownik jest zalogowany |
| Scenariusz główny | 1) Użytkownik wybiera przycisk „profil” z paska bocznego  2) System wyświetla dane profilu  3) Użytkownik naciska przycisk „edytuj dane”  4) System zmienia sposób wyświetlania danych tak, by dało się je edytować  5) Po wprowadzeniu zmian, użytkownik naciska przycisk „zapisz”  6a) System zapisuje wprowadzone zmiany |
| Scenariusz alternatywny | 6b) System informuje użytkownika o niekompletności lub nieprawidłowości danych i prosi o ich uzupełnienie |
| Po zakończeniu | System zmienia sposób wyświetlania danych, tak by nie dało się ich edytować i wyświetlany jest komunikat o pomyślnej aktualizacji profilu. |

|  |  |
| --- | --- |
| Przypadek użycia | **6. Zmiana hasła** |
| Aktor | Użytkownik |
| Cel | Zmiana hasła służącego do logowania |
| Warunki wstępne | Użytkownik jest zalogowany |
| Scenariusz główny | 1) Użytkownik wybiera przycisk „profil” z paska bocznego  2) System wyświetla dane profilu  3) Użytkownik naciska przycisk „zmień hasło”  4) System wyświetla formularz zmiany hasła  5) Użytkownik wprowadza obecne i nowe hasło  6) Użytkownik naciska przycisk „zmień hasło”  7a) System zapisuje wprowadzone zmiany |
| Scenariusz alternatywny | 7b) System informuje użytkownika o niekompletności lub nieprawidłowości danych i prosi o ich uzupełnienie |
| Po zakończeniu | Wyświetlany jest komunikat o pomyślnej aktualizacji hasła. |

|  |  |
| --- | --- |
| Przypadek użycia | **7. Usunięcie konta** |
| Aktor | Użytkownik |
| Cel | Usunięcie konta użytkownika z systemu wraz ze wszystkimi powiązanymi z nim informacjami |
| Warunki wstępne | Użytkownik jest zalogowany |
| Scenariusz główny | 1) Użytkownik wybiera przycisk „profil” z paska bocznego  2) System wyświetla dane profilu  3) Użytkownik naciska przycisk „usuń konto”  4) System wyświetla formularz usuwania konta  5) Użytkownik wprowadza hasło  6) Użytkownik naciska przycisk „usuń konto”  7a) System usuwa konto użytkownika |
| Scenariusz alternatywny | 7b) System informuje użytkownika o braku lub nieprawidłowości hasła i prosi o jego podanie |
| Po zakończeniu | Użytkownik zostaje przeniesiony na stronę główną przed zalogowaniem, gdzie wyświetlany jest komunikat o pomyślnym usunięciu konta. |

### 3.3.2 Lista PU modułu hodowlanego

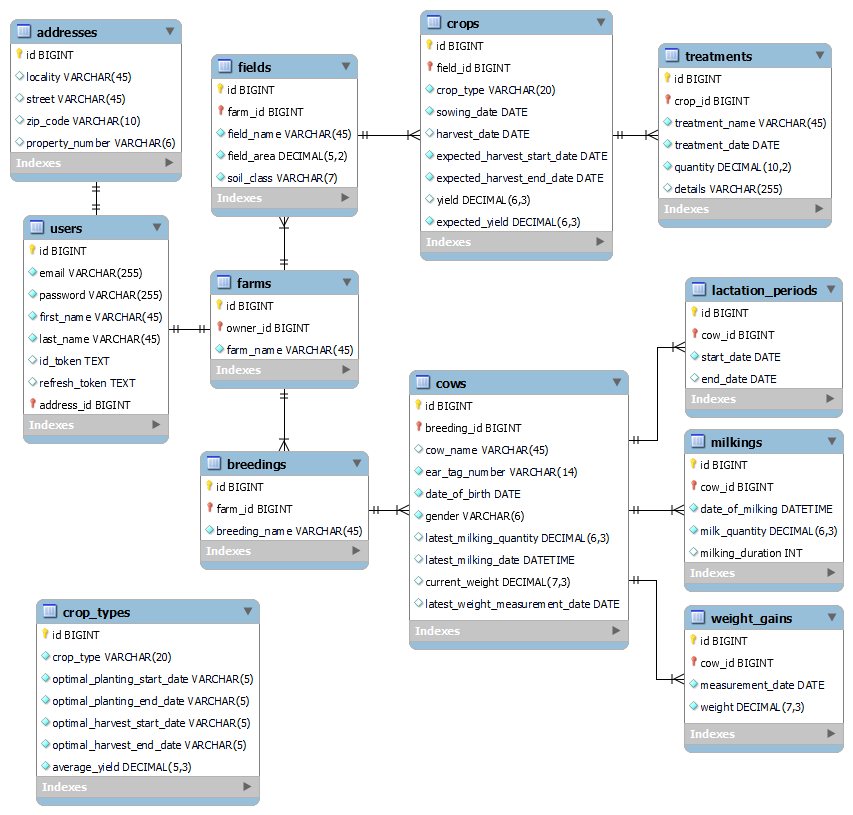
Wszystkie przedstawione PU w tej sekcji, dotyczyć będą dziedziny **8. Zarządzanie hodowlą (Moduł hodowlany)**. Zakładać będą one również, że użytkownik jest już zalogowany.

|  |  |
| --- | --- |
| Przypadek użycia | **8.1.2 Dodanie stada** |
| Aktor | Użytkownik |
| Cel | Dodanie nowego stada do systemu |
| Warunki wstępne | Użytkownik jest na stronie „hodowla” |
| Scenariusz główny | 1a) Użytkownik naciska przycisk „dodaj stado”  2) System wyświetla formularz dodawania stada  3) Użytkownik wprowadza nazwę stada  4) Użytkownik naciska przycisk „dodaj”  5a) System dodaje nowe stado |
| Scenariusz alternatywny | 1b) Użytkownik posiada już stado, naciska przycisk z dużym plusem (który po najechaniu wyświetla podpis „dodaj stado”) 5b) System informuje użytkownika nieprawidłowości nazwy i prosi o jej poprawę |
| Po zakończeniu | System wyświetla widok nowoutworzonego stada oraz wyświetla informację o pomyślnym dodaniu stada. |

# Rozdział IV. Schemat bazy danych

## 4.1 Wprowadzenie i schemat

Aby przybliżyć sposób działania aplikacji od strony technicznej, w niniejszym rozdziale szczegółowo omówiony zostanie schemat bazy danych, która jest fundamentem systemu, pełniąc rolę centralnego repozytorium do przechowywania i zarządzania danymi. W ramach aplikacji „FarmGuider”, baza danych została zaprojektowana tak, aby wspierać zarządzanie gospodarstwem rolnym, oferując struktury danych dedykowane zarówno dla hodowli bydła, jak i upraw rolnych. Schemat bazy danych prezentuje logiczną strukturę tych danych – tabele, ich pola (kolumny), typy danych, a także związki między nimi.



## 4.2 Tabele użytkownika i strategia identyfikacji rekordów

W przypadku aplikacji „FarmGuider”, każda tabela zawiera kolumnę id, która jest wykorzystywana jako klucz główny. Id to unikalny identyfikator, który służy do jednoznacznego rozróżnienia każdego rekordu w tabeli. Identyfikator ten jest typu bigint, który jest typem danych liczbowych, umożliwiającym przechowywanie bardzo dużych wartości liczbowych. Wybór typu bigint dla tej kolumny, jest decyzją strategiczną, która antycypuje przyszłe wymagania systemu w kontekście skalowania. Taki typ danych został wybrany z myślą o możliwości obsługi znacząco rosnącej liczby rekordów, które mogą się pojawić wraz ze wzrostem liczby użytkowników aplikacji i związanej z tym akumulacji danych w bazie.

Baza danych „FarmGuider” używa sekwencji SQL do generowania unikalnych wartości dla kolumny id. Jest to instrukcja, która tworzy sekwencję liczbową, rozpoczynającą się od liczby 1000 i zwiększającą się o 1 z każdym nowym rekordem. Użycie sekwencji zapewnia, że każde nowe id jest unikalne, co jest istotne dla integralności danych i ich relacyjności.

W opisach tabel, kolumna id nie jest listowana, ze względu na jej powtarzalny charakter. Została ona powyżej opisana jako uniwersalne pole, przyjmując założenie, że każda tabela w bazie danych posiada tę kolumnę, służącą jako klucz główny.

* Tabela **users** – służąca przechowywaniu danych użytkownika, ale także kluczowa dla bezpieczeństwa – na podstawie jej pól użytkownik jest autentykowany i autoryzowany oraz określana jest ważność sesji. Zawiera następujące kolumny:
  + **email** – kolumna przechowująca adres użytkownika, zawierająca unikatowe wartości. Jest ona typu varchar(255), co oznacza ciąg znaków o maksymalnej długości 255 znaków, co jest standardowym rozmiarem adresów email. Pole te nie może być puste.
  + **password** – przechowuje hash hasła użytkownika, który podczas próby zautentykownia jest porównywany, z hashem hasła wprowadzonego podczas próby. Typ kolumny password to varchar(255). Pole te nie może być puste.
  + **first\_name –** pole typu varchar(45), przechowujące imię użytkownika. Długość 45 znaków jest wystarczająca by pomieścić większość imion. Pole te nie może być puste.
  + **last\_name –** kolumna przechowująca nazwisko użytkownika, również typu varchar(45). Pole te nie może być puste.
  + **id\_token** – służy do przechowywania JWT – tokenu identyfikującego sesję. Jest typu text, wykorzystywanego do dłuższych ciągów znaków, mogących przekroczyć ograniczenia typu varchar.
  + **refresh\_token** – kolumna typu text, przechowująca inny niż id\_token typ JWT. JWT przechowywany w tej kolumnie służy do odświeżenia wygasłej sesji.
  + **address\_id** – klucz obcy typu bigint, odnoszący się do tabeli addresses. Relacja z tabelą addresses jest typu 1:1, co oznacza, że jeden użytkownik ma przypisany do siebie jeden adres w systemie.
* Tabela **addresses** – służąca do przechowywania danych adresowych użytkownika. Wszystkie pola w tej tabeli są opcjonalne, a są nimi:
  + **locality** – kolumna typu varchar(45), służąca do przechowywania nazwy miejscowości zamieszkania.
  + **street** – pole typu varchar(45), zawierające nazwę ulicy.
  + **zip\_code** – służy do przechowywania kodu pocztowego, jest typu varchar(10).
  + **property\_number –** pole typu varchar(6), przechowujące numer budynku.
* Tabela **farms** – tabela stworzona z myślą o skalowalności systemu, tak by w przyszłości bez trudu dało się dodać opcję posiadania przez użytkownika więcej niż jednego gospodarstwa (bądź by użytkownik mógł być przypisany do gospodarstwa, którego nie jest właścicielem). Ponadto jest to tabela pełniąca rolę pośrednika, między tabelą users, a tabelami przeznaczonymi dla modułu uprawowego i hodowlanego. Tabela zawiera następujące kolumny:
  + **farm\_name** – kolumna typu varchar(45), zawierająca nazwę gospodarstwa. Pole te nie może być puste i jest uzupełniane automatycznie na podstawie id użytkownika.
  + **owner\_id** – klucz obcy typu bigint, odnoszący się do tabeli users. Pole te zawiera id właściciela gospodarstwa. Wpis w tabeli farms tworzony jest automatycznie podczas rejestracji użytkownika w systemie. Relacja z tabelą users jest typu 1:1, co oznacza, że jeden użytkownik może posiadać tylko jedno gospodarstwo.

## 4.3 Tabele modułu uprawowego

* Tabela **fields** – służąca do przechowywania danych, na temat pól uprawnych gospodarstwa użytkownika. Odgrywa zasadniczą rolę w organizowaniu i zarządzaniu informacjami dotyczącymi modułu uprawowego. Na tabelę składają się następujące kolumny:
  + **field\_name** – kolumna typu varchar(45), zawierająca nazwę pola uprawnego, które w systemie służy do łatwiejszej identyfikacji pól, przez użytkownika. Nazwa pola nie może być pusta.
  + **field\_area** – reprezentuje powierzchnię pola wyrażoną w hektarach, co jest istotne w kwestii zapisu historii plonów oraz przewidywania plonów przyszłych. Typ kolumny to decimal(5,2), służący do przechowywania wartości liczbowych, gdzie 5 oznacza ilość znaków liczby, a 2 oznacza ilość znaków przeznaczonych na wartości ułamkowe. Pole te nie może być puste.
  + **soil\_class** – kolumna przechowująca tekstowe wartości enumeryczne, dotyczące klas gruntu. Jest to pole wymagane. W przypadku, gdy użytkownik nie poda tej informacji, rejestrowany jest status oznaczający, że klasa gleby nie została określona. Pole jest typu varchar(7).
  + **farm\_id** – klucz obcy typu bigint, odnoszący się do tabeli farms. Pole te zawiera id gospodarstwa. Relacja z tabelą farms jest typu 1:n, co oznacza, że jedno gospodarstwo może posiadać wiele pól.
* Tabela **crop\_types** – służąca do przechowywaniaw systemie informacji o typach dostępnych upraw. Uzupełniana jest przez developera, a każde pole w tej tabeli jest wymagane. Nie łączy się przez relację z żadną inną tabelą. Jej pola, służące do analizy upraw w gospodarstwach, to:
  + **crop\_type** – kolumna, która przechowuje tekstowe wartości enumeryczne, informujące o typie danej uprawy. Na podstawie tego pola, identyfikowane są uprawy w systemie. Pole jest typu varchar(20).
  + **optimal\_planting\_start\_date** – kolumna przechowująca informacje o dacie rozpoczęcia optymalnego okresu siewnego danej uprawy. Jest typu varchar(5), ponieważ wprowadzone dane powinny być w formacie MM-DD, gdzie MM oznacza miesiąc, a DD – dzień miesiąca. Jest to spowodowane wymogiem uniwersalności danych dla każdego roku.
  + **optimal\_planting\_end\_date** – kolumna przechowująca informacje o dacie zakończenia optymalnego okresu siewnego danej uprawy. Podobnie jak optimal\_planting\_start\_date, jest typu varchar(5).
  + **optimal\_harvest\_start\_date** - kolumna przechowująca informacje o dacie rozpoczęcia optymalnego okresu zbioru danej uprawy. Podobnie jak optimal\_planting\_start\_date, jest typu varchar(5).
  + **optimal\_harvest\_end\_date** - kolumna przechowująca informacje o dacie zakończenia optymalnego okresu zbioru danej uprawy. Podobnie jak optimal\_planting\_start\_date, jest typu varchar(5).
  + **average\_yield** – pole przechowujące informacje na temat średniego plonu w tonach, z hektara danej uprawy. Na jego podstawie obliczany jest przewidywany plon. Jest to pole typu decimal(5,3).
* Tabela **crops** – służąca do przechowywania informacji o historii upraw na danym polu. Składa się z następujących pól:
  + **crop\_type** – przechowuje te same tekstowe wartości enumeryczne, informujące o typie danej uprawy, które są przechowywane przez kolumnę crop\_type w tabeli crop\_types. Jest to identyfikator, na podstawie którego, określa i pobiera się typ uprawy do analizy. Tak samo jak owa kolumna w crop\_types, jest to wymagane pole typu varchar(20).
  + **sowing\_date** – kolumna przechowująca datę siewu danej uprawy. Jest to pole wymagane, typu date. Typ date oznacza datę.
  + **harvest\_date** – kolumna przechowująca datę zbioru danej uprawy. Pole te nie jest wymagane, ale powinno zostać uzupełnione po zbiorze uprawy. Typ kolumny to date.
  + **expected\_harvest\_start\_date** – wymagana kolumna typu date, zawierająca informację na temat daty rozpoczynającej okres przewidywanego, optymalnego zbioru. Jest obliczana na podstawie kolumny optimal\_harvest\_start\_date z tabeli crop\_types. Została dodana do crops z dwóch powodów. Pierwszy z nich to powód optymalizacyjny, by nie pobierać za każdym razem danych z tabeli crop\_types w celu obliczenia tego pola. Drugi zaś to zapis archiwalny, gdyby po jakimś czasie optimal\_harvest\_start\_date zostało edytowane, to pole expected\_harvest\_start\_date nie ulegnie zmianie.
  + **expected\_harvest\_end\_date** – wymagana kolumna typu date, zawierająca informację na temat daty kończącej okres przewidywanego, optymalnego zbioru. Jest obliczana na podstawie kolumny optimal\_harvest\_end\_date z tabeli crop\_types. Została dodana z tych samych powodów co expected\_harvest\_start\_date.
  + **yield** – pole przechowujące informacje o zebranym z pola plonie w tonach. Podobnie do harvest\_date, pole te nie jest wymagane, ale powinno zostać uzupełnione po zbiorze uprawy. Jest to pole typu decimal(6,3).
  + **expected\_yield** – kolumna zawierająca informacje na temat przewidywanego plonu z pola, w tonach. Wymagana wartość typu decimal(6,3) obliczana jest na podstawie average\_yield z crop\_types. Pole te zostało dodane z tych samych powodów co expected\_harvest\_start\_date.
  + **field\_id** - klucz obcy typu bigint, odnoszący się do tabeli fields. Kolumna ta zawiera id pola uprawnego. Relacja z tabelą fields jest typu 1:n, co oznacza, że jedno pole może posiadać wiele upraw.
* Tabela **treatments** - służąca do przechowywania informacji o historii zabiegów stosowanych na danej uprawie. Składa się z następujących pól:
  + **treatment\_name** – kolumna zawierająca informacje o typie/nazwie zastosowanego na uprawie zabiegu. Jest to wymagane pole typu varchar(45).
  + **treatment\_date** – pole zawierające datę wykonanego zabiegu. Informacja ta jest wymagana, a jej typ to date.
  + **quantity** – służy do przechowywania informacji o ilości zastosowanego środka. Kolumna typu decimal(10,2), która jest wymagana.
  + **details** – opcjonalne pole zawierające informacje dodatkowe, które użytkownik chciałby dołączyć do informacji o zastosowanym zabiegu, środku. Jest to pole typu varchar(255).

## 4.4 Tabele modułu hodowlanego

* Tabela **breedings** - służąca do przechowywania danych, na temat stad zwierząt w gospodarstwie użytkownika. Jej rolą jest organizowanie informacji dotyczących hodowli. Na tabelę składają się następujące kolumny:
  + **breeding\_name** – kolumna przechowująca nazwę stada. Pole typu varchar(45), które jest wymagane.
  + **farm\_id** – klucz obcy typu bigint, odnoszący się do tabeli farms. Pole te zawiera id gospodarstwa. Relacja z tabelą farms jest typu 1:n, co oznacza, że jedno gospodarstwo może posiadać wiele stad.
* Tabela **cows** – służąca do przechowywania danych, na temat krów w gospodarstwie użytkownika. Odgrywa zasadniczą rolę w agregowaniu informacji dotyczących modułu hodowlanego. Na tabelę składają się następujące kolumny:
  + **cow\_name** – opcjonalne pole typu varchar(45), zawierające imię krowy.
  + **ear\_tag\_number** – kolumna zawierająca numer kolczyka krowy, po którym można łatwo, jednoznacznie zidentyfikować zwierzę. Jest to pole wymagane, typu varchar(14), co jest standardowym rozmiarem numeru kolczyka.
  + **date\_of\_birth –** służy do przechowywania informacji o dacie urodzenia zwierzęcia. Pole wymagane typu date.
  + **gender** - kolumna, która przechowuje tekstowe wartości enumeryczne, informujące o płci zwierzęcia. Na jego podstawie blokowane są niektóre funkcjonalności systemu dotyczące płci. Pole jest wymagane i jest typu varchar(6).
  + **latest\_milking\_quantity** – pole przechowujące ilość litrów mleka najnowszego udoju. Kolumna ta została dodana w celach optymalizacyjnych, by przy odczycie rekordu zwierzęcia czy odczycie listy zwierząt, rekordy dotyczące ostatniego udoju nie musiały być wyszukiwane i dołączane do wyniku. Pole te jest typu decimal(6,3) i nie jest wymagane.
  + **latest\_milking\_date** – kolumna zawierająca datę wraz z czasem dla najnowszego udoju. Podobnie do kolumny latest\_milking\_quantity, ta kolumna również została dodana w celach optymalizacyjnych. Pole te nie jest wymagane i jest typu datetime, służącego do przechowywania daty oraz czasu.
  + **current\_weight** – zawiera wartość najnowszego pomiaru wagi mierzonej w kilogramach. Podobnie do pola latest\_milking\_quantity, ta kolumna również została dodana w celach optymalizacyjnych. Kolumna ta jest typu decimal(7,3) i nie jest wymagana.
  + **latest\_weight\_measurement\_date** – przechowuje date najnowszego pomiaru wagi. Podobnie do kolumny latest\_milking\_quantity, ta kolumna również została dodana w celach optymalizacyjnych. Pole te jest typu date i nie jest wymagane.
  + **breeding\_id** – klucz obcy typu bigint, odnoszący się do tabeli breedings. Pole te zawiera id stada. Relacja z tabelą breedings jest typu 1:n, co oznacza, że jedno stado może posiadać wiele krów.
* Tabela **lactation\_periods** – służąca do przechowywania danych, na temat okresów laktacyjnych krowy. Na podstawie okresu laktacyjnego mierzona jest wydajność mleczna zwierzęcia. Na tabelę składają się następujące kolumny:
  + **start\_date** – kolumna przechowująca datę rozpoczęcia okresu laktacyjnego krowy. Typ tej kolumny, której uzupełnienie jest wymagane, to date.
  + **end\_date** – pole zawierające datę zakończenia okresu laktacyjnego. Pole te jest typu date i nie wymaga podania. Kiedy pole end\_date jest puste, okres laktacyjny traktowany jest jako obecnie trwający.
  + **cow\_id** – klucz obcy typu bigint, odnoszący się do tabeli cows. Pole te zawiera id krowy. Relacja z tabelą cows jest typu 1:n, co oznacza, że jedna krowa może posiadać wiele okresów laktacyjnych.
* Tabela **milkings** – służąca do przechowywania danych o udojach krowy. Na podstawie danych z niniejszej tabeli, generowany jest raport wydajności mlecznej krowy. Tabela zawiera następujące kolumny:
  + **date\_of\_milking** – kolumna przechowująca informację o dacie udoju wraz z czasem. Kolumna ta wymaga podania, a jej typ to datetime.
  + **milk\_quantity** – pole zawierające informację na temat ilości mleka w litrach, uzyskanego podczas udoju. Pole te jest obligatoryjne, a jego typ to decimal(6,3).
  + **milking\_duration** – służy do przechowywania czasu udoju mierzonego w sekundach. Pole te jest opcjonalne, a jego typ to int, który służy do przechowywania wartości liczbowych (ale nie z tak dużego zakresu jak bigint).
  + **cow\_id** – klucz obcy typu bigint, odnoszący się do tabeli cows. Pole te zawiera id krowy. Relacja z tabelą cows jest typu 1:n, co oznacza, że jedna krowa może posiadać wiele udojów.
* Tabela **weight\_gains** – jest to tabela, która przechowuje dane na temat pomiarów wagi bydła. W oparciu o dane z tej tabeli, tworzony jest wykres zmian masy ciała zwierzęcia. Tabela zawiera następujące kolumny:
  + **measurement\_date** – kolumna zawierająca datę pomiaru wagi zwierzęcia. Typ kolumny to date i wymaga ona uzupełnienia.
  + **weight** – pole przechowujące wagę uzyskaną podczas pomiaru wagi. Waga powinna być wyrażona w kilogramach. Typ pola to decimal(7,3) i jest to pole wymagane.

# Rozdział V. Przewodnik użytkownika po aplikacji