****

**System rezerwacji dla stadniny koni**

*Praca projektowa z przedmiotu*

*Modelowanie i Analiza Systemów Informatycznych*

Prowadzący: Autor/zy:

mgr inż. Ewa Żesławska Damian Jamroży, 113729

Patryk Arendt, 113705

Rzeszów 2024

Spis treści

[1 Opis założeń projektu 3](#_Toc173149236)

[2 Modelowanie systemu i jego otoczenia 5](#_Toc173149237)

[3 Opis struktury projektu 12](#_Toc173149238)

[4 Prezentacja warstwy użytkowej projektu 15](#_Toc173149239)

[5 Podsumowanie 24](#_Toc173149240)

1. Opis założeń projektu

Projekt nosi nazwę "System do Zarządzania Stadniną Koni". Jego celem jest stworzenie kompleksowego systemu informatycznego, który umożliwi zarządzanie rezerwacjami wizyt, końmi, trenerami oraz klientami w stadninie koni. Kontekst biznesowy projektu opiera się na wspomaganiu właścicieli stadnin w codziennych operacjach, usprawnieniu procesów rezerwacji i zarządzania zasobami oraz poprawie jakości usług świadczonych klientom. System będzie dostępny jako strona internetowa, wykorzystując technologie takie jak PHP, HTML, CSS, JavaScript oraz bazę danych MySQL. Aplikacja umożliwi użytkownikom rezerwowanie wizyt, a administratorom zarządzanie końmi, trenerami oraz harmonogramem rezerwacji.

Celem projektu jest stworzenie łatwego w użyciu, niezawodnego i efektywnego systemu zarządzania stadniną koni, który zautomatyzuje procesy administracyjne i rezerwacyjne. Zakres projektu obejmuje tworzenie i zarządzanie profilami klientów, zarządzanie rezerwacjami wizyt, zarządzanie informacjami o koniach i trenerach, tworzenie harmonogramów treningów i dostępności, interfejs administracyjny do zarządzania wszystkimi aspektami stadniny oraz interfejs użytkownika do rezerwacji wizyt.

Problemem, który będzie rozwiązywany, są obecne metody zarządzania rezerwacjami i zasobami w stadninach koni, które są często manualne, czasochłonne i podatne na błędy. Brak zintegrowanego systemu powoduje chaos w zarządzaniu harmonogramami, co może prowadzić do podwójnych rezerwacji, nieefektywnego wykorzystania zasobów oraz niezadowolenia klientów. Podstawowym źródłem problemu jest brak dedykowanego systemu informatycznego, który centralizowałby wszystkie informacje i procesy związane z zarządzaniem stadniną.

Rozwiązanie problemu jest kluczowe dla poprawy efektywności operacyjnej stadniny, zwiększenia zadowolenia klientów oraz optymalizacji wykorzystania zasobów. Dowody na istnienie problemu obejmują opinie właścicieli stadnin, wskazujące na trudności w zarządzaniu harmonogramami i rezerwacjami, analizę przypadków podwójnych rezerwacji i ich wpływ na działalność stadniny oraz dane dotyczące niskiego poziomu automatyzacji procesów w stadninach.

Niezbędne zasoby do rozwiązania problemu obejmują zespół programistów z doświadczeniem w technologiach webowych (PHP, HTML, CSS, JavaScript) oraz bazach danych (MySQL), analizę wymagań biznesowych i specyfikację funkcjonalną, środowisko deweloperskie i testowe oraz stały kontakt z właścicielami stadnin w celu zbierania wymagań i testowania aplikacji. Te zasoby są kluczowe, aby zapewnić właściwe zrozumienie wymagań biznesowych, zaprojektowanie i implementację funkcjonalnego systemu oraz przetestowanie go pod kątem użyteczności i niezawodności.

Realizacja projektu będzie przebiegać krok po kroku, zaczynając od analizy wymagań, która obejmuje zbieranie i analizę wymagań biznesowych od właścicieli stadnin oraz tworzenie specyfikacji funkcjonalnej systemu. Następnie nastąpi projektowanie, które obejmuje projektowanie interfejsu użytkownika (UI) i doświadczenia użytkownika (UX) oraz architektury systemu. Implementacja będzie polegała na tworzeniu bazy danych MySQL, implementacji backendu w PHP, implementacji frontendu w HTML, CSS, JavaScript oraz integracji wszystkich komponentów systemu. Kolejnym krokiem będzie testowanie, obejmujące testy jednostkowe, integracyjne i akceptacyjne oraz zbieranie opinii od właścicieli stadnin i użytkowników końcowych. Po zakończeniu testów nastąpi wdrożenie systemu na serwer produkcyjny, konfiguracja serwera WWW i bazy danych. Ostatnim etapem będzie szkolenie właścicieli stadnin i użytkowników końcowych oraz tworzenie dokumentacji użytkownika i administracyjnej. Wynikiem prac będzie w pełni funkcjonalna aplikacja internetowa do zarządzania stadniną koni, umożliwiająca rezerwację wizyt, zarządzanie końmi, trenerami i klientami oraz usprawnienie codziennych operacji administracyjnych w stadninie.

1. Modelowanie systemu i jego otoczenia

Aby lepiej zrozumieć funkcjonalności dostępne w aplikacji utworzony oraz przedstawiony został diagram przypadków użycia.

Obraz zawierający tekst, czarne i białe, diagram, szkic

Opis wygenerowany automatycznie

Rysunek 1 Diagram przypadków użycia

Na diagramie zdefiniowano kilka typów aktorów:

* Klient – aktor, który może korzystać z kilku funkcjonalności. Jest to zalogowany użytkownik ma wgląd w swoje dane oraz może tworzyć nowe rezerwacje.
* Trener – aktor, który podobnie jak klient ma jedynie kilka funkcjonalności, dodatkowo może akceptować rezerwacje, które są przypisane do niego. Dodatkowo aktor ten może przeglądać wszystkich klientów w aplikacji oraz edytować dane konia.
* Administrator – aktor, który ma wszystkie uprawnienia. Ma dostęp do praktycznie wszystkich funkcjonalności, może edytować, dodawać oraz usuwać wszystkie rekordy takie jak konie, użytkownicy itp. Aktor ten ma wgląd we wszystkie rezerwacje w systemie może je anulować lub dodać nową.

Do niektórych z przedstawionych przypadków użycia na diagramie została utworzona tabelka scenariuszów użycia. W tabeli znajdują się informacje co zostanie wykonane w danym przypadku oraz kroki przez które trzeba przejść, aby ukończyć poszczególny przypadek.

|  |  |
| --- | --- |
| Lp. | PU-1 |
| Nazwa: | Dodanie nowej rezerwacji |
| Aktorzy: | Gość |
| Warunki początkowe: | Aktor zalogował się |
| Opis: | Aktor chce dodać nową rezerwację |
| Ścieżka główna | 1. Aktor przechodzi do terminarza. 2. Aktorowi zostaje wyświetlony terminarz z jego rezerwacjami oraz guzikiem do dodania nowej rezerwacji 3. Aktor klika przycisk ‘Nowa Rezerwacja’ 4. Aktor wypełnia wszystkie pola. 5. Aktor zatwierdza wpisane dane poprzez kliknięcie przycisku „Zapisz”. |
| Ścieżka alternatywna: | - |
| Warunki końcowe: | Aktor dodał nową rezerwację. |

Tabela 1 Scenariusz użycia nr 1

|  |  |
| --- | --- |
| Lp. | PU-2 |
| Nazwa: | Anulowanie rezerwacji |
| Aktorzy: | Gość |
| Warunki początkowe: | Aktor zalogował się |
| Opis: | Aktor chce anulować istniejącą rezerwację |
| Ścieżka główna | 1. Aktor przechodzi do terminarza. 2. Aktorowi zostaje wyświetlony terminarz z jego rezerwacjami oraz guzikiem do dodania nowej rezerwacji 3. Aktor klika na terminarzu rezerwację którą chce anulować. 4. Aktorowi wyświetla się podgląd rezerwacji. 5. Aktor klika w guzik „Anuluj rezerwację” |
| Ścieżka alternatywna: | 3a. Aktor klika w dzień, w którym ma rezerwację, którą chce anulować.  3b. Aktor klika w rezerwację którą chce anulować. |
| Warunki końcowe: | Aktor anulował rezerwację. |

Tabela 2 Scenariusz użycia nr 2

|  |  |
| --- | --- |
| Lp. | PU-3 |
| Nazwa: | Zaakceptowanie rezerwacji |
| Aktorzy: | Trener |
| Warunki początkowe: | Aktor zalogował się |
| Opis: | Aktor chce zaakceptować rezerwację. |
| Ścieżka główna | 1. Aktor przechodzi do terminarza. 2. Aktorowi zostaje wyświetlony terminarz z jego rezerwacjami oraz guzikiem do dodania nowej rezerwacji. 3. Aktor klika na terminarzu rezerwację którą chce zaakceptować. 4. Aktorowi wyświetla się podgląd rezerwacji. 5. Aktor klika w guzik „Zaakceptuj rezerwację” |
| Ścieżka alternatywna: | 3a. Aktor klika w dzień, w którym ma rezerwację, którą chce zaakceptować.  3b. Aktor klika w rezerwację którą chce zaakceptować |
| Warunki końcowe: | Aktor pomyślnie zaakceptował rezerwację. |

Tabela 3 Scenariusz użycia nr 3

|  |  |
| --- | --- |
| Lp. | PU-4 |
| Nazwa: | Dodanie nowego użytkownika |
| Aktorzy: | Administrator |
| Warunki początkowe: | Aktor zalogował się |
| Opis: | Aktor chce dodać nową rezerwację |
| Ścieżka główna | 1. Aktor przechodzi do ‘Dane osobowe’. 2. Aktorowi zostaje wyświetlona tabela, z wszystkimi użytkownikami, oraz guzikiem do tworzenia nowego użytkownika. 3. Aktor klika przycisk ‘Dodaj uzytkownika’. 4. Aktor wypełnia wszystkie pola. 5. Aktor zatwierdza wpisane dane poprzez kliknięcie przycisku „Zapisz”. |
| Ścieżka alternatywna: | - |
| Warunki końcowe: | Aktor pomyślnie dodał nowego użytkownika. |

Tabela 4 Scenariusz użycia nr 4

Diagram Aktywności

Bazowym procesem w całej aplikacji jest dodanie nowej rezerwacji oraz zapis jej w bazie danych. Poniższy diagram przedstawia przebieg komunikowania się użytkownika z aplikacją w celu dodania nowej rezerwacjiObraz zawierający tekst, zrzut ekranu, diagram, projekt graficzny

Opis wygenerowany automatycznie

Rysunek 2 Diagram aktywności

Przedstawiona czynność przy swojej realizacji wykorzystuje kilka osobnych elementów takich jak:

* Front-end -> użytkownik
* Backend -> Aplikacja internetowa
* Baza danych

Użytkownik chcąc dodać nową rejestrację musi być zalogowany, następnie przejść na stronę z terminarzem a następnie dodać nową rejestrację rozpoczynając cały proces. W formularzu po dodaniu daty w której użytkownik chce dodać rezerwację pojawiają się do wyboru trenerzy oraz konie, które są dostępne w danym terminie. Po poprawnym uzupełnieniu wszystkich danych, są one wysyłane na serwera bazodanowego, który tworzy nowy rekord w bazie danych, a następnie zwraca informację o sukcesie lub błędzie. Aplikacja odbiera informację o błędzie lub sukcesie i zwraca ją do klienta. Po zakończeniu tego procesu w terminarzu będzie widoczna nowa rezerwacja. Rezerwacja będzie posiadała status ‘oczekujacy’ co oznacza, że rezerwacja jest w systemie i oczekuje na akceptację przez wybranego trenera.

Diagram sekwencji

Obraz zawierający zrzut ekranu, tekst, diagram, design

Opis wygenerowany automatycznie

Rysunek 3 Diagram sekwencji

Diagram przedstawia cały proces dodania nowej rezerwacji do bazy, z wyszczególnieniem całego przepływu procesu. Całość rozpoczyna się od użytkownika, który dodaje nową rezerwację, następnie przeglądarka przetwarza dodanie, serwer wysyła zapytanie do bazy danych, która odbiera zapytanie oraz przetwarza je. Następnie dodaje nowy rekord do bazy danych oraz zwraca informację o sukcesie lub błędzie do serwera. Serwer poprzez przeglądarkę wyświetla rezultat użytkownikowi.

Diagram klas

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

Rysunek 4 Diagram klas

Diagram przedstawia klasy w projekcie oraz ich metody. Większość rzeczy wykonywanych jest na bazie danych, dlatego poszczególne klasy nie posiadają metod.

1. Opis struktury projektu

Aplikacja ta wykorzystuje języki webowe takie jak PHP, HTML, CSS, JavaScript. Aby uruchomić aplikację należy uprzednio wgrać bazę danych zamieszczoną do projektu pod nazwą klub\_jezdziecki, a następnie uruchomić aplikację w środowisku serwerowym, który umożliwi poprawne wykonanie plików PHP w wersji 8 np. xampp.

**Minimalne wymagania sprzętowe:**

**Serwer (hosting współdzielony lub VPS):**

* **Procesor:** 1 rdzeń CPU (np. Intel Xeon lub AMD Opteron)
* **Pamięć RAM:** 1 GB
* **Dysk twardy:** 10 GB wolnej przestrzeni (preferowany dysk SSD)
* **Łącze internetowe:** 1 Gbps

**Oprogramowanie:**

* **System operacyjny:** Linux (np. Ubuntu, CentOS) lub Windows Server
* **Serwer WWW:** Apache lub Nginx
* **PHP:** Wersja 7.4 lub nowsza
* **MySQL:** Wersja 5.7 lub nowsza
* **SSL/TLS:** Certyfikat SSL dla bezpiecznego połączenia HTTPS

**Zalecane wymagania sprzętowe:**

**Serwer (VPS lub dedykowany):**

* **Procesor:** 2 rdzenie CPU (np. Intel Xeon lub AMD EPYC)
* **Pamięć RAM:** 2-4 GB
* **Dysk twardy:** 20-50 GB wolnej przestrzeni (preferowany dysk SSD)
* **Łącze internetowe:** 1 Gbps

**Oprogramowanie:**

* **System operacyjny:** Linux (np. Ubuntu, CentOS) lub Windows Server
* **Serwer WWW:** Apache lub Nginx z konfiguracją optymalizacyjną dla PHP
* **PHP:** Wersja 7.4 lub nowsza (z modułami: mysqli, pdo\_mysql)
* **MySQL:** Wersja 5.7 lub nowsza (preferowany MariaDB dla lepszej wydajności)
* **SSL/TLS:** Certyfikat SSL (np. Let's Encrypt lub komercyjny)

**Dodatkowe zalecenia:**

* **Kopia zapasowa:** Regularne tworzenie kopii zapasowych bazy danych i plików strony
* **Monitoring:** Monitorowanie zasobów serwera (CPU, RAM, dysk, ruch sieciowy)
* **Bezpieczeństwo:** Regularne aktualizacje oprogramowania, firewall, skanowanie pod kątem złośliwego oprogramowania

**Uwagi:**

* **Optymalizacja:** W celu zwiększenia wydajności, warto zastosować optymalizacje bazy danych oraz kodu PHP. Można rozważyć użycie cache (np. Memcached, Redis) do przechowywania często używanych danych.
* **Skalowalność:** Jeśli strona ma obsługiwać dużą ilość użytkowników, warto zastanowić się nad skalowalnymi rozwiązaniami, takimi jak load balancing, replikacja bazy danych, czy serwery aplikacji.

Te wymagania zapewnią, że strona będzie działała płynnie i bez większych problemów, obsługując rezerwacje wizyt w stadninie koni.

W systemie zarządzania stadniną koni, dane będą przechowywane i zarządzane za pomocą relacyjnej bazy danych MySQL. Struktura bazy danych została zaprojektowana w taki sposób, aby umożliwić łatwe zarządzanie informacjami o koniach, rezerwacjach, klientach i trenerach. Baza danych składa się z kilku głównych tabel, takich jak horses, clients, trainers, reservations, oraz tabel pomocniczych jak horses\_breed, horses\_color, health\_status, horse\_type, które przechowują szczegółowe informacje o różnych aspektach zarządzania stadniną. Tabela horses zawiera takie pola jak id, imie, kolor, rasa, stan\_zdrowia, rodzaj\_konia, opis, data\_urodzenia, wzrost, zdjecie, natomiast tabela clients przechowuje informacje o klientach, takie jak id, imie, nazwisko, email, telefon. Tabela trainers zawiera dane trenerów, w tym id, imie, nazwisko, specjalizacja, a tabela reservations przechowuje informacje o rezerwacjach, takie jak id, id\_client, id\_horse, id\_trainer, data\_rezerwacji, status.

Projekt zakłada stworzenie hierarchii klas w aplikacji, które będą odpowiadały za zarządzanie danymi w systemie. Klasa Horse będzie odpowiadała za zarządzanie danymi dotyczącymi koni i będzie miała takie pola jak id, imie, kolor, rasa, stan\_zdrowia, rodzaj\_konia, opis, data\_urodzenia, wzrost, zdjecie, z metodami CRUD (Create, Read, Update, Delete) dla tabeli horses. Klasa Client będzie odpowiadała za zarządzanie danymi klientów, posiadając pola id, imie, nazwisko, email, telefon oraz metody CRUD dla tabeli clients. Klasa Trainer będzie odpowiadała za zarządzanie danymi trenerów, mając pola id, imie, nazwisko, specjalizacja oraz metody CRUD dla tabeli trainers. Klasa Reservation będzie odpowiadała za zarządzanie rezerwacjami, posiadając pola id, id\_client, id\_horse, id\_trainer, data\_rezerwacji, status oraz metody CRUD dla tabeli reservations.

Obraz zawierający tekst, diagram, Plan, Równolegle

Opis wygenerowany automatycznie

Rysunek 5 Diagram ERD

1. Prezentacja warstwy użytkowej projektu

Przykładowe dane logowania, na konta z istniejącymi uprawnieniami:

|  |  |  |  |
| --- | --- | --- | --- |
| **Rola** | Administrator | Trener | Klient |
| **Login** | admin@admin | trener@trener | klient@klient |
| **Hasło** | admin123 | trener@trener | klient@klient |

Tabela 5 Dane logowania

Strona startowa aplikacji zawarta jest w pliku Index.php w katalogu sites.

Na wskazanej powyżej stronie mamy możliwość logowania się na już istniejące konto. Pola na wskazanej stronie są walidowane i weryfikowane z zawartością w bazie danych. Jeżeli użytkownik istnieje, system sprawdza jaką ma przypisaną rolę do konta, a następnie przekierowuje go na stronę Dashboard.php z odpowiednimi uprawnieniami do odczytu dalszych zakładek.

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Strona internetowa

Opis wygenerowany automatycznie

Rysunek 6 Index.php

Po zalogowaniu się na każde z kont, niezależnie od ustawionych uprawnień widzimy identyczny podgląd strony startowej. Użytkownik widzi wszystkie dostępne dla niego zakładki w systemie.

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Ikona komputerowa

Opis wygenerowany automatycznie

Rysunek 7 Dashboard.php

Po przejściu do zakładki Trenerzy użytkownik otrzymuje listę osób z rolą Trener. Jeżeli użytkownik posiada pełne uprawnienia, czyli jego rola w systemie to administrator, wówczas ukazują mu się dodatkowe kolumny takie jak edytuj oraz usuń, a także przycisk na dole pod nazwą Dodaj trenera. Jeżeli użytkownik posiada rolę Trenera, wówczas dodatkowo widzi jedynie przycisk Edycji przy wierszu ze swoimi danymi, natomiast w przypadku klienta, widzi on jedynie zdjęcie, imię, nazwisko oraz stopień jeździecki, bez możliwości wprowadzania jakichkolwiek zmian. W widoku tym zastosowano rozwiązanie filtrowania oraz sortowania. Aby wyfiltrować wybrane elementy wystarczy, że wpiszemy imię lub nazwisko, które nas interesuje, a następnie przyciśniemy enter, lub opuścimy pole wyszukiwarki. Działa ona dynamicznie i zmienia wyświetlone rekordy w zależności od wpisanych wartości. Stopień jeździecki można filtrować poprzez wybranie odpowiedniej wartości z listy rozwijanej. Sortowanie działa w dwie strony, zatem możemy sortować rosnąco jak i malejąco, aby dokonać sortowania wystarczy, że klikniemy na nazwę wybranej kolumny. Zarówno zakładki jak i działania takie jak filtrowanie oraz sortowanie działa bez przeładowywania całej strony. Zamieniana jest jedynie zawartość wybranego elementu, dzięki funkcji dynamicznych adresów URL.

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Strona internetowa

Opis wygenerowany automatycznie

Rysunek 8 downloat\_trainers.php

Jako administrator strony możemy skorzystać z funkcji CRUD. Aby dodać nowego użytkownika klikamy przycisk „Dodaj trenera”. Wykonanie tej czynności wywoła modal, który zawiera formularz z danymi, które administrator musi uzupełnić. Aby usunąć użytkownika, wystarczy, że administrator kliknie na przycisk Usuń, po prawej strony wybranego wiersza. Edycja danych jest analogiczna do dodawania nowego trenera. Użytkownik otrzymuje modal, który jednocześnie nakłada warstwę przyciemniającą na ekran, tak aby formularz był bardziej widoczny. Oprócz podstawowych informacji, strona zawiera również obsługę zdjęć. Użytkownik, może wskazać lokalizację zdjęcia na komputerze, lub przeciągnąć je bezpośrednio na pole drag & drop. Wykonanie takiej czynności spowoduje wczytanie zdjęcia do formularza, a po jego przesłaniu, zdjęcie zostaje skopiowane i wgrane na dysk serwerowy, w tym przypadku do katalogu img/employee. W przypadku edycji, zdjęcie oraz wszelkie informacje automatycznie uzupełniają formularz na podstawie danych zawartych w bazie danych. W przypadku edycji którejkolwiek z danych, informacje zostają przesłane do bazy danych, a zawartość odpowiednich tabel zostaje nadpisana. Zdjęcia użytkownika w bazie przechowywane są za pomocą ścieżki do pliku. System, w przypadku usunięcia użytkownika, usuwa także jego fizyczne zdjęcie z systemu, w przypadku edycji, działa to analogicznie. Stara fotografia zostaje usunięta, a nowa wgrana na serwer.

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

Rysunek 9 download\_trainers.php - Edycja danych (strona zmniejszona do 70%)

Zakładka Konie działa analogicznie, jednakże w tym przypadku administrator posiada pełną obsługę CRUD, trener może jedynie edytować konia, przy czym może zmieniać tylko jego stan zdrowotny, natomiast klient widzi tylko podgląd danych bez możliwości dokonywania jakichkolwiek zmian. Również tutaj zastosowane zostały filtry oraz sortowania. Zdjęcia dla koni przechowywane są w katalogu img/horses.

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Ikona komputerowa

Opis wygenerowany automatycznie

Rysunek 10 download\_horse.php

Obraz zawierający tekst, oprogramowanie, zrzut ekranu, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

Rysunek 11 download\_horse.php - Edycja danych (strona zmniejszona do 70%)

Tak samo jak w poprzednich dwóch zakładkach, zakładka Dane osobowe wyświetla tabelę z informacjami odnośnie użytkowników. W tym przypadku informacji tych jest więcej. W tabeli zawarte są wszystkie dane osobowe wprowadzone do systemu. Widok ten widoczny jest jedynie dla administratorów oraz trenerów, przy czym tylko administrator może wprowadzać jakiekolwiek zmiany. Dodatkową funkcjonalnością w tym widoku jest możliwość zmiany roli w systemie dla wybranego użytkownika. Jeżeli użytkownik, poprzednio miał rolę Trener lub Administrator to jego zdjęcie istniało zapisane w katalogu img/employee, w przypadku zmiany roli na klient, zdjęcie zostaje przeniesione do katalogu img/users. Funkcja ta działa w każdą stronę, w przypadku zmiany roli na trener lub administrator zdjęcie zostaje przeniesione do img/employee. Dodatkowo widoczność dla trenera jest ograniczona, w tabeli widzi on jedynie klientów, bez wglądu do administratorów czy tez innych trenerów.

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Ikona komputerowa

Opis wygenerowany automatycznie

Rysunek 12 client\_data\_All.php - Widok administratora

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Ikona komputerowa

Opis wygenerowany automatycznie

Rysunek 13 client\_data\_All.php - Widok trenera

Jeżeli rola użytkownika to klient, wtedy otrzymuje on inne przekierowanie. Zamiast zakładki Dane osobowe, widzi on zakładkę Moje dane, które pozwalają mu jedynie na podgląd oraz edycję własnych danych zawartych w systemie.

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Strona internetowa

Opis wygenerowany automatycznie

Rysunek 14 client\_data.php - Widok klienta

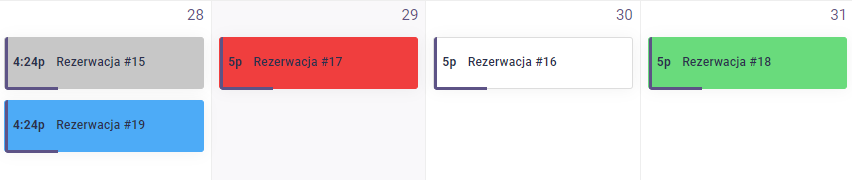
Ostatnim widokiem, jest widok Terminarza. To w nim odbywa się najważniejsza część systemu czyli umawianie wizyt. Kalendarz jest współdzielony dla wszystkich użytkowników, jednakże tyko administrator widzi wszystkie rezerwacje. Trener oraz klient widzą tylko te, które są do nich przypisane.

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Ikona komputerowa

Opis wygenerowany automatycznie

Rysunek 15 terminarz.php

Kalendarz posiada trzy rodzaje podglądu: miesięczny, tygodniowy oraz dzienny. Klikając w poszczególne przyciski możemy się między nimi przełączać. Dodatkowo w przypadku kliknięcia na wybrany dzień, zostajemy przeniesieni do podglądu tego dnia. W systemie istnieją trzy rodzaje rezerwacji. Odwołana, oczekująca na akceptację oraz aktywna.



Rysunek 16 Terminarz - Rodzaje rezerwacji

Za rodzaje oraz statusy rezerwacji odpowiadają kolory:

* Niebieski – Rezerwacja, która się już odbyła, zakończona pomyślnie
* Szary – Rezerwacja przeszła, odwołana
* Czerwony – Rezerwacja odwołana
* Biały – Rezerwacja oczekująca na akceptację
* Zielony – Rezerwacja przyjęta i zaakceptowana

W zakładce terminarz tylko klienta lub administrator może dodawać rezerwację poprzez formularz. Administrator ma ten sam formularz co klient, dodatkowo posiada on także do wybrania klienta tak aby dało się go powiązać z trenerem. Pola posiadają walidację, konie oraz trenerzy wyświetlają się tylko Ci którzy są dostępni w wybranym terminie. W innym przypadku nie ma ich dostępnych na liście.

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Ikona komputerowa

Opis wygenerowany automatycznie

Rysunek 17 Terminarz - Dodawanie rezerwacji

Po dokonaniu rezerwacji, trener może potwierdzić lub odrzucić wybraną rezerwację, która została do niego przypisana.

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

Rysunek 18 Terminarz - Akceptacja rezerwacji

1. Podsumowanie

Projekt „System do Zarządzania Stadniną Koni” ma na celu stworzenie kompleksowego, zautomatyzowanego systemu zarządzania rezerwacjami wizyt, końmi, trenerami oraz klientami w stadninie koni. System ten, zaprojektowany jako aplikacja internetowa, wykorzystuje technologie takie jak PHP, HTML, CSS, JavaScript oraz bazę danych MySQL, aby zapewnić łatwe w użyciu, niezawodne i efektywne narzędzie wspomagające codzienne operacje w stadninie.

Projekt zakłada centralizację wszystkich informacji i procesów, co jest kluczowe dla poprawy efektywności operacyjnej, zwiększenia zadowolenia klientów oraz optymalizacji wykorzystania zasobów. Dowody na istnienie problemów w obecnych metodach zarządzania, takie jak trudności w zarządzaniu harmonogramami i przypadki podwójnych rezerwacji, potwierdzają potrzebę stworzenia takiego systemu. Wdrożenie systemu ma na celu wyeliminowanie tych problemów poprzez automatyzację i centralizację zarządzania informacjami o rezerwacjach, koniach, klientach i trenerach.

Hierarchia klas w aplikacji została zaprojektowana tak, aby odpowiadała za zarządzanie różnymi aspektami systemu. Klasa Horse zarządza danymi dotyczącymi koni, klasa Client zarządza danymi klientów, klasa Trainer zarządza danymi trenerów, a klasa Reservation zarządza danymi rezerwacji. Każda z tych klas zawiera metody CRUD (Create, Read, Update, Delete), co umożliwia pełne zarządzanie danymi w systemie.

Diagram przepływu danych (DFD) ilustruje, jak informacje przepływają między różnymi komponentami systemu. Użytkownik wprowadza dane rezerwacji, które są przetwarzane przez warstwę logiki biznesowej i zapisywane w tabeli reservations. Administratorzy mogą zarządzać danymi koni, klientów i trenerów, a system generuje raporty i analizy na podstawie zebranych danych, prezentując je administratorowi w formie graficznej lub tabelarycznej.

Realizacja projektu przebiega etapami, począwszy od analizy wymagań, przez projektowanie interfejsu użytkownika i architektury systemu, implementację backendu i frontendu, testowanie, wdrożenie, aż po szkolenie użytkowników końcowych i tworzenie dokumentacji. Wynikiem prac jest w pełni funkcjonalna aplikacja internetowa, która znacząco usprawnia codzienne operacje administracyjne w stadninie, umożliwiając efektywne zarządzanie rezerwacjami wizyt, końmi, trenerami i klientami. System ten przyczyni się do zwiększenia satysfakcji klientów, poprawy organizacji pracy oraz optymalizacji wykorzystania zasobów w stadninie koni.