

**POLITECHNIKA KRAKOWSKA IM. TADEUSZA KOŚCIUSZKI**  
**W KRAKOWIE**

**Wydział Inżynierii Elektrycznej i Komputerowej**

**Kierunek Informatyka**

**PRACA INŻYNIERSKA**

***Zintegrowany system zarządzania projektem informatycznym***

Piotr Adam Tomaszewski

**Promotor:**

dr hab. inż. Mieczysław Drabowski

**Konsultant:**

mgr inż. Dariusz Dorota

**KRAKÓW 2016**

Kropla drąży skałę nie siłą, lecz często padając.

Owidiusz

# Spis treści

1. Wstęp.....	5
2. Opis problemu .....	7
2.1 Zarządzanie projektem informatycznym.....	7
2.2 Najpopularniejsze aplikacje do zarządzania projektami .....	8
2.2.1 Trello .....	8
2.2.2 JIRA.....	8
2.2.3 MS Project.....	9
3. Opis produktu.....	10
3.1 Wizja aplikacji.....	10
3.2 Określenie perspektywy administratora aplikacji.....	11
3.3 Określenie perspektywy administratora baz danych aplikacji.....	11
3.4 Określenie perspektywy użytkownika.....	12
3.5 Zastosowane technologie, biblioteki i narzędzia .....	12
3.5.1 Front-end systemu .....	12
3.5.2 Back-end systemu.....	14
3.5.3 Zintegrowane środowisko programistyczne .....	15
4. Analiza wymagań systemu .....	17
4.2 Wymagania niefunkcjonalne .....	18
4.3 Scenariusze użytkownika .....	19
4.3.1 Logowanie do systemu .....	19
4.3.2 Zmiana hasła.....	20
4.3.3 Tworzenie nowego projektu .....	20
4.3.4 Dodawanie nowego programisty .....	21
4.3.5 Tworzenie zadania do wykonania .....	22
4.3.6 Kontakt z administratorem .....	22
5. Modelowanie systemu .....	24
5.1 Modelowanie granic systemu .....	24
5.2 Modelowanie struktury baz danych.....	25
5.3 Modelowanie procesów.....	26
5.3.1 Modelowanie komunikacji systemu z obiektami .....	26
5.3.2 Modelowanie głównych funkcjonalności.....	27
5.4 Modelowanie zdarzeń.....	29
5.5 Modelowanie przepływu informacji.....	31

6. Projekt bazy danych .....	35
6.1 Określenie atrybutów encji.....	36
6.2 Ważniejsze polecenia języka SQL .....	37
6.3 Skrypty języka SQL .....	38
7. Dokumentacja techniczna dla administratora aplikacji i baz danych.....	39
8. Dokumentacja techniczna dla użytkowników .....	44
8.1 Logowanie do systemu .....	44
8.2 Rejestracja do systemu .....	45
8.3 Panel użytkownika .....	46
8.4 Zmiana danych osobowych lub hasła.....	46
8.5 Tworzenie projektu.....	47
8.6 Praca nad projektem .....	48
8.7 Obserwowanie projektu.....	49
8.8 Zapraszanie pracowników i klientów do projektu.....	49
8.9 Tworzenie zadań w projekcie.....	51
8.10 Kontakt z administratorem .....	52
9. Podsumowanie.....	53
9.1 Testy manualne.....	54
9.2 Możliwości rozwoju .....	54
9.3 Zakończenie.....	55
Bibliografia.....	56
Netografia.....	57
Spis ilustracji .....	58
Spis tabel .....	59

# 1. Wstęp

Informatyka ciągle się rozwija, można powiedzieć, że coraz bardziej dynamicznie. Co rok poznajemy nowe technologie, nowe języki programowania oraz nowe platformy pozwalające budować aplikację. Każdego roku pojawiają się również firmy wykorzystujące innowacyjne technologie oraz zatrudniające coraz większą ilość pracowników, w tym programistów, testerów i grafików. Chcąc osiągać zyski firma musi szukać klientów, a im lepsza renoma firmy tym łatwiejsza droga dotarcia do klienta. Aby taką renomę osiągnąć należy pozyskać odpowiednią kadrę i umiejętnie nią zarządzać. W dzisiejszych czasach firma pracuje na wielu projektach jednocześnie, co sprawia, że zarządzanie nimi i osobami koordynującymi nimi jest znacznie utrudnione. Niepewność czy uda się wyrobić w terminie, presja czasu i wypadki losowe spowodowały, że w firmach, tych dużych jak i tych małych, zaczęto stosować rozwiązania ułatwiające zarządzanie osobami i zadaniami związanymi z projektem, stając się istotnym zagadnieniem. Umiejętność pracy w grupie, z zadaniami które należy wykonać w określonym czasie jest szczególnie istotna w branży IT.

Tematem mojej pracy inżynierskiej jest właśnie zagadnienie dotyczące zarządzania projektem informatycznym. Już na etapie nauki na Politechnice Krakowskiej wraz z innymi studentami zetknęliśmy się z problemem pracy w grupie. Problemem o dziwo nie była komunikacja pomiędzy nami, a podział zadań do wykonania, biorąc pod uwagę umiejętności i doświadczenie poszczególnych osób. Udało nam się znaleźć porozumienie wykorzystując narzędzia dostępne on-line. Aplikacje takie jak Trello, JIRA czy MS Project były nawet odpowiednie dla naszych projektów, jednak zawsze im czegoś brakowało. Jedne były zbyt proste i banalne, nieposiadające potrzebnych funkcjonalności. Inne zaś zbyt obszerne w udogodnienia, których i tak nie udawało nam się wykorzystać w pełni.

Aplikacja, która jest wykonywana w ramach pracy inżynierskiej ma za zadanie połączyć prostotę aplikacji, które wykorzystaliśmy z doбором odpowiednich funkcjonalności, pozwalających na wystarczającą kontrolę nad projektem. Dodatkowym urozmaicheniem będzie możliwość kontroli projektu przez klienta. Osoba będąca klientem będzie miała wgląd w bieżącą pracę programistów, również sama będzie mogła decydować czy wykonana funkcjonalność spełnia oczekiwania czy nie. Jest to bardzo pomocne, ponieważ pozwala na bieżącą edycję aplikacji do potrzeb klienta. Bez tej funkcjonalności programista musiałby znaleźć moduł, który pisał np. kolega i go poprawić.

W kolejnych częściach pracy zostaną przedstawione wszystkie zagadnienia związane z założeniami, tworzeniem, administracją, obsługą oraz rozwojem aplikacji. W drugim rozdziale umieszczono zagadnienia dotyczące problematyki zarządzania projektami informatycznymi. Omówiono dlaczego to jest tak ważne i czemu coraz częściej wymagana jest przez pracodawców umiejętność dostosowania się do zadań. Przedstawiono pokrótce najpopularniejsze aplikacje webowe i programy desktopowe do zarządzania.

Rozdział 3 przedstawia wizję, aplikacji do której dążono. Chodzi głównie o wprowadzenie panelu dla klienta, który na bieżąco ma podgląd w pracę programistów pracujących przy projekcie. Omówiono perspektywę każdej osoby, która występuje. Przedstawione zostały również technologie i modele wykorzystywane do stworzenia systemu.

W rozdziale 4 omówiono analizę wymagań systemu, czyli przedstawiono wymagania funkcjonalne i нефункционалне wraz z najważniejszymi scenariuszami użytkowników.

Rozdział 5 to przedstawienie graficzne całego systemu, tj. zestawienie wszelkich możliwych diagramów. Zaprezentowano diagram przedstawiający tak zwanych aktorów i związki pomiędzy nimi w określonej dziedzinie przedmiotowej. Diagram związków encji przedstawiający graficzną wizualizację baz danych. Diagram przepływu danych z diagramem kontekstowym i systemowym oraz diagram historii życia encji.

W kolejnym rozdziale, omówiono zagadnienia dotyczące bazy danych. Przedstawiono proces jej projektowania, określono relację między encjami wraz z krotnością oraz atrybutami, przedstawiono klucze główne i obce. Pokazano więzy integralności z przykładowymi poleceniami SQL.

Rozdział 7 jest przeznaczony dla administratorów, którzy chcieliby zainstalować taki system na własnym serwerze. Ten rozdział jest dość konkretną instrukcją pozwalającą w szybki i łatwy sposób uporać się z zadaniem.

Rozdział 8 jest instrukcją obsługi portalu dla zwykłego użytkownika. Pokazuje ciekawe rozwiązania implementacyjne, instrukcje prezentujące np. działania na zadaniach. Opisuje symbole nawigacji i możliwe odnośniki lub przyciski.

W ostatnim, 9 rozdziale, dotyczącym aplikacji, przedstawiono podsumowanie projektu. Zawarto informację dla osób, które chciałyby rozwinąć tę aplikację o dodatkowe moduły albo funkcjonalności poprawiając wygląd i sprawność systemu.

## 2. Opis problemu

Czas to pieniądź – to powiedzenie na pierwszy rzut oka, wydaje się dość prymitywne i oczywiste, ale gdyby się tak zastanowić można wywnioskować, że posiada głębsze przesłanie. Dzięki obrotowi pieniądza funkcjonuje rozwój gospodarki, ludzie mogą zarabiać, rozwijać swoje umiejętności, zlecać zadania, odpoczywać. Takie zagadnienie jest również istotne w branży informatycznej, kiedy klient zleca projekt do wykonania, a w głowie Project Ownera, czy managera leży kwestia określenia czasu wykonania i kosztów z tym związanych.

### 2.1 Zarządzanie projektem informatycznym

Wytwarzanie programów jest jednym wielkim procesem produkcji. Widać to, zwłaszcza, podczas kończącego się terminu wyznaczonego do stworzenia jakiejś aplikacji, bądź jej wdrożenia. To co jest znane i zdefiniowane przecież łatwiej kontrolować. Można również przewidzieć czas zakończenia prac, a co za tym idzie koszt. Problemy przy wytwarzaniu oprogramowania związane są z niewiedzą lub spornością odnośnie wymagań postawionych przez klienta. Dlatego dąży się do usystematyzowania pracy.

W jednym z modeli systemu informatycznego, a dokładnie w modelu kaskadowym, inaczej zwanym liniowym lub tradycyjnym, projektowanie znajduje się na pozycji 3 za analizą potrzeb i specyfikacją systemu. W tym modelu zakłada się, że na początku każdego projektu istnieją konkretnie zdefiniowane potrzeby określone przez klienta. Ważną uwagą jest fakt, że te procesy nie zmieniają się w trakcie życia systemu. W modelu Fry'ego faza projektowania zawiera w sobie analizę potrzeb z opisem danych i procesów oraz projektowanie fizyczne, określające strukturę zbiorów, pamięci i wzorców dokumentów. Model ten dotyczy przede wszystkim projektowania systemów z zastosowaniem baz danych.

Przy tworzeniu oprogramowania pojawia się bardzo często jakiś problem. Może być związany z projektem, załogą pracowników, klientem lub gospodarką. Dlatego należy podjąć ryzyko, które jest jednym z ważniejszych procesów. Takie decyzje podejmowane są przez osobę kompetentną. W trakcie podejmowania go możemy przewidzieć lub nie, skutki jakie dana decyzja może spowodować. Możliwe jest także przewidzenie ryzyka w oparciu o prawdopodobieństwo wystąpienia wydarzeń, na podstawie dostępnych informacji. Nie jest to łatwa sprawa, ponieważ może spowodować straty i wpłynąć negatywnie na projekt.

## **2.2 Najpopularniejsze aplikacje do zarządzania projektami**

Każdym projektem należy odpowiednio zarządzać. W zależności od dostępnych narzędzi możemy przyjąć koncepcję prostego gospodarowania projektem przy użyciu karteczek lub przy użyciu skomplikowanych narzędzi, dających szeroki wachlarz możliwości.

### **2.2.1 Trello**

Pierwszym z omawianych narzędzi jest Trello. To tak naprawdę startup, stworzony przez firmę FogCreek, której twarzą jest Joel Spolsky – znany amerykański bloger i programista, którego znak rozpoznawczy to jakość tworzonych aplikacji.

Jednak czym jest tak naprawdę Trello? Otóż, jest to aplikacja internetowa, dostępna z poziomu przeglądarki, również tej na urządzeniach mobilnych, służąca do zarządzania zadaniami. Wyobraźmy sobie, że mamy w domu tablicę podzieloną na kilka kolumn, w których określamy przykładowo przestrzeń zaplanowanych, wykonywanych i ukończonych zadań. Każde zadanie jakie sobie zaplanujemy to jedna samoprzylepna karteczka. Możemy także określić priorytet i osobę, która miałaby wykonać zadanie. Na warunki domowe jest to dość niewygodne i bardzo kosztowne, ponieważ karteczki szybko się zużywają, a chcąc na przykład założyć nową tablicę z zupełnie innym zarządzaniem, to mogło by nie wystarczyć miejsca. Dlatego z pomocą przychodzi Trello, aplikacja dostępna z każdego miejsca, pod warunkiem, że użytkownik ma dostęp do Internetu. Możemy tworzyć dowolną ilość tablic, na których dodawane są zadania poprzez przeciągnięcie i upuszczenie (drag-and-drop). Można je w łatwy sposób sortować, grupować, określać priorytet, komentować, ustalać termin zakończenia zadania, co w warunkach domowych byłoby bardzo trudne.

Aplikacja jest bardzo prosta w użyciu. Przydaje się, gdy niepotrzebne nam są jakiekolwiek integracje z innymi systemami. Również na wypadek zbliżania się końca terminu wykonania, wysyłana jest przypominająca wiadomość na adres e-mail, ale nie tylko. Użytkownik powiadamiany jest również o dyskusjach, nowych zadaniach i projektach.

### **2.2.2 JIRA**

Kolejne środowisko, w porównaniu do poprzedniego jest znacznie rozbudowane. Zapewnia ono nie tylko zarządzanie zadaniami, lecz również osobami jak i ogólnie projektami. Narzędzie stworzone przez firmę Atlassian, wykorzystujące założenie programowania zwinnego, w którym klient, w każdym momencie, może zmienić swoje wymagania.



Narzędzie pozwala w bardzo dokładny sposób na określenie typu pracownika (czy jest to na przykład programista, tester, analityk), na oznaczenie kto jest twórcą zadania i komu zleca się jego wykonanie. Istnieje również możliwość określenia terminu rozpoczęcia i zakończenia danego zadania, jak i również uzależnienia jednego zadania od drugiego. Twórcy aplikacji zadbali o graficzny interfejs, aby pomimo bogatej funkcjonalności był prosty i domyślny w użyciu.

Przydatny jest przede wszystkim osobom zarządzającym projektami: managerom, projekt leaderom. Jednak i zwykli developerzy pracujący nad zleceniem w zespołach też mogą skorzystać używając JIRA.

### **2.2.3 MS Project**

Microsoft Project to kolejna aplikacja ceniona w środowisku małych i średnich firm mających problem z zarządzaniem i koordynowaniem projektów, biorąc pod uwagę zarządzanie pracownikami i budżetem. Jest to narzędzie desktopowe, podobne w swojej strukturze do MS Word lub MS Excel.

Daje możliwość planowania i nadzorowania zadań, budowy harmonogramów i konstruowania raportów. Managerowie otrzymują konkretny przegląd sytuacji, kto jakie zadanie ma do zrobienia, a programiści – szybkie zapoznanie się z zaplanowanym dla niego zadaniem i łatwe odnalezienie się w projekcie. Jest to narzędzie firmy Microsoft, stąd wizualizacja diagramów czy innych schematów nie powinna dziwić, ponieważ wykorzystują te same modele graficzne, które również wykorzystywane są w programach takich jak MS Excel czy MS PowerPoint.

Ciekawymi funkcjonalnościami w tym narzędziu jest definiowanie zadań w sposób hierarchiczny. Polega to na tym, że określam dowolnie duże zadanie, które posiada mniejsze podzadania, rozdzielone na różnych programistów i różne terminy zakończenia. Po określeniu takich zależności, narzędzie jest w stanie narysować harmonogram w postaci Wykresu Gantta.

Twórcy dali możliwość importu lub eksportu projektu. Narzędzie, jak inne tego typu programy, nie są darmowe, a jedyna bezpłatna wersja to wersja testowa trwająca przez 30 dni. Kolejnym mankamentem jest fakt, iż MS Project nie jest dostępny w polskiej wersji językowej, stąd osoby zaczynające pracę w branży IT, a nieznające zbyt dobrze języka angielskiego mogą mieć problemy z zaawansowaną terminologią.

### **3. Opis produktu**

Po zapoznaniu się z najpopularniejszymi dostępnymi narzędziami na rynku, zrodził się we mnie pewien niedosyt, co sprawiło, że stworzyłem aplikację, która łączy w sobie pewne funkcjonalności w jedno wraz z nowymi właściwościami, z którymi nie spotkałem się do tej pory.

#### **3.1 Wizja aplikacji**

Pierwszym z założeń, była chęć stworzenia aplikacji uniwersalnej dla wszystkich. Niezależnie czy byłby to projekt związany z informatyką czy też nie. Taka charakterystyka została zaczerpnięta z idei aplikacji Trello, którą każdy, w dowolny sposób może wykorzystywać do własnych celów (np. zarządzania domem, koordynowania projektem). Myśl jest słuszna, ponieważ nie ma sensu skupiać się tylko na jednym rynku docelowym. Im więcej odbiorców aplikacji, tym lepiej.

Kolejnym założeniem to prostota użytkowania. Nieważne, czy będzie to osoba młodego pokolenia czy doświadczona latami – każda powinna wiedzieć jak obsługiwać aplikację, przemieszczać się od strony do strony, wysyłać wiadomości czy określać zadania. Interfejs użytkownika powinien być jasny i przejrzysty. Bez zbędnych animacji, reklam, niepotrzebnych odnośników. Jednak zawsze, w razie jakichkolwiek kłopotów, użytkownik będzie mógł skorzystać z dokumentacji użytkownika lub kontaktu z administratorem w celu pomocy lub zgłoszenia ewentualnego błędu.

Oprócz wizualizacji graficznej całej aplikacji ważna też jest funkcjonalność. Dlatego skupiono się na konstruowaniu zadań w sposób hierarchiczny. Taka prezentacja projektu polegająca na konstruowaniu podzadań jest bardzo trafna i przyspiesza pracę. Nad jednym zadaniem skupia się jeden programista w ramach jego zdolności. Nie dostaje zadania, z którym nie mógłby sobie poradzić. Stąd tak ważny jest dobór pracowników. Pracownik ma określony czas na wykonanie zadania. Jeśli wykona go, zgłasza do sprawdzenia i jeśli jest wszystko w porządku, to funkcjonalność zostaje zatwierdzona przez menagera.

Ostatnią i najważniejszą wizją aplikacji jest możliwość wglądu prac przez klienta. Chodzi o to, że klient ma prawo, na bieżąco obserwować poziom prac, które są wykonywane. Dostaje on możliwość stwierdzenia czy funkcjonalność, którą wykonał programista i którą potwierdził manager jest zgodna z jego założeniami.

### **3.2 Określenie perspektywy administratora aplikacji**

Zadania administratora aplikacji są następujące:

- Prawidłowa komunikacja pomiędzy bazą danych, aplikacją a użytkownikiem
- Dostosowanie aplikacji webowej pod różne urządzenia komputerowe i mobilne
- Zwiększanie bezpieczeństwa aplikacji przed wszelkimi atakami
- Stworzenie struktury aplikacji, w celu przejrzystego interfejsu graficznego
- Ciągła dbałość o wydajność aplikacji
- Rozwiązywanie i dokumentowanie problemów związanych z użytkowaniem

Administrator wykonuje aktualizację w plikach znajdującej się na serwerze, dlatego niezbędna jest przez niego znajomość zasad i terminologii tworzenia stron internetowych, przede wszystkim języków: HTML5, CSS3, JavaScript, PHP.

### **3.3 Określenie perspektywy administratora baz danych aplikacji**

Zadania administratora baz danych są następujące:

- Zabezpieczenie baz przed utratą danych
- Tworzenie hurtowni danych do celów statystycznych dla twórców aplikacji
- Dbłość o zachowanie odpowiednich relacji pomiędzy encjami
- Konstruowanie właściwych zapytań MySQL zleconych przez twórców aplikacji
- Troska o komunikację pomiędzy bazą danych a aplikacją

Administrator wykonuje aktualizację w bazie danych znajdującej się na serwerze, dlatego niezbędna jest przez niego znajomość zasad i terminologii w relacyjnych bazach danych, również zasad konstrukcji i zarządzania hurtownią danych.

### **3.4 Określenie perspektywy użytkownika**

Zadania użytkownika są następujące:

- Możliwość rejestracji do systemu zarządzania, zmiany danych osobowych i hasła
- Stworzenie nowego projektu z określeniem nazwy, daty rozpoczęcia i zakończenia
- Przydział użytkowników i klientów do projektu, wyznaczenie i przegląd zadań
- Potwierdzanie wykonania funkcjonalności przez programistę
- Komunikacja pomiędzy użytkownikami w projekcie,

### **3.5 Zastosowane technologie, biblioteki i narzędzia**

Docelową ideą tworzonego systemu zarządzania projektem informatycznym jest dostępność z poziomu przeglądarki internetowej. Pomimo skomplikowanej aplikacji użytkowanie będzie wymagało urządzenia z zainstalowaną przeglądarką i z dostępnym połączeniem do Internetu.

#### **3.5.1 Front-end systemu**

W związku z tym, że jest to aplikacja internetowa nie obejdzie się bez języków takich jak HTML, CSS oraz JavaScript. Bez ich użycia strony nie miałyby jakiegokolwiek wizualnego sensu, a użytkowanie byłoby znacząco utrudnione i niewygodne.

HTML jest tak zwanym językiem znaczników. Chodzi o to, że tekst dodawany na stronę jest dodatkowo interpretowany właśnie poprzez wykorzystanie znaczników – znaków napisanych wewnątrz nawiasów kątowych. Można określić czy ma to być akapit, formularz, nagłówek albo link. Odpowiednie obszary w HTML określają części przeznaczone dla przeglądarki, wyszukiwarki i odbiorcy zawierające informacje o zawartości pomiędzy znacznikami. Język ten posiada poza znacznikami, również atrybuty, które pozwalają np. na komunikację z kaskadowym arkuszem stylów. Istnieje również odmiana HTML, która jest bardziej strukturalna: XHTML. Od momentu rozpoczęcia tworzenia stron internetowych było już kilka wersji HTML, natomiast aktualna wersja języka to HTML5, będący rozwinięciem HTML4 i XHTML. W ostatnim czasie powstało dużo programów, które pomagają i przyspieszają tworzenie stron www laikom jak i profesjonalistom, poprzez autouzupełnianie znaczników lub gotowe nazwy atrybutów.

Obok HTML, który bardziej opisuje strukturę wyświetlania strony, istnieją kaskadowe arkusze stylów CSS, które reprezentują wizualną część strony internetowej. Wszystkie kwestie dotyczące grafiki, rodzaju i rozmiaru czcionek, zmiany koloru tła przycisku i innych zagadnień obsługuje właśnie CSS. Kaskadowe arkusze stylów posiadają reguły, które określają sposób wyświetlania konkretnych znaczników i elementów. Reguła taka składa się z selektora i deklaracji, zaś deklaracja zbudowana jest z właściwości i wartości oddzielonych od siebie dwukropkiem. Dobrym nawykiem jest umieszczanie reguł CSS w osobnych plikach, ale istnieje możliwość dołączenia sekcji stylów w pliku dokumentu HTML.

Sam HTML i CSS pozwala zwizualizować już w jakiś sposób stronę. Jednak będzie ona statyczna, bez animacji, interakcji z użytkownikiem. Aby nadać życie naszej stronie wykorzystuje się do tego JavaScript. Tym razem jest to język skryptowy wykorzystywany na stronach WWW. Dzięki niemu możemy zbudować użytkownikowi wygodny dla niego interfejs, co nie jest prostym zadaniem. W tym, o którym mowa w pracy inżynierskiej, zamiast natywnego języka JavaScript wykorzystano bibliotekę JQuery. Jest to najpopularniejszy framework dostępny w środowisku informatycznym. Biblioteka pozwala szybciej i z mniejszą ilością kodu pisać skrypty. Największą zaletą jest fakt, że istnieje kompatybilność z różnymi przeglądarkami. Prosta składnia, biblioteka minimalnych rozmiarów i ogromna ilość wtyczek powodują, że skrypty wykonują się błyskawicznie, co jest dużym atutem w porównaniu do mniej znanych frameworków.

Nieomawianym jeszcze dotąd narzędziem, a dość często wykorzystywanym w systemie od strony wizualnej jest framework CSS: Bootstrap. Stworzony przez twórców Twittera posiada gotowe funkcjonalności HTML i CSS pozwalające w łatwy sposób kreować graficzny layout strony również wyświetlając ją z dostosowaniem się do urządzeń mobilnych. Dzięki temu narzędziu tworzenie stron jest znacznie szybsze, gdyż nie trzeba tworzyć od zera reguł dostosowanych pod urządzenia przenośne.

### 3.5.2 Back-end systemu

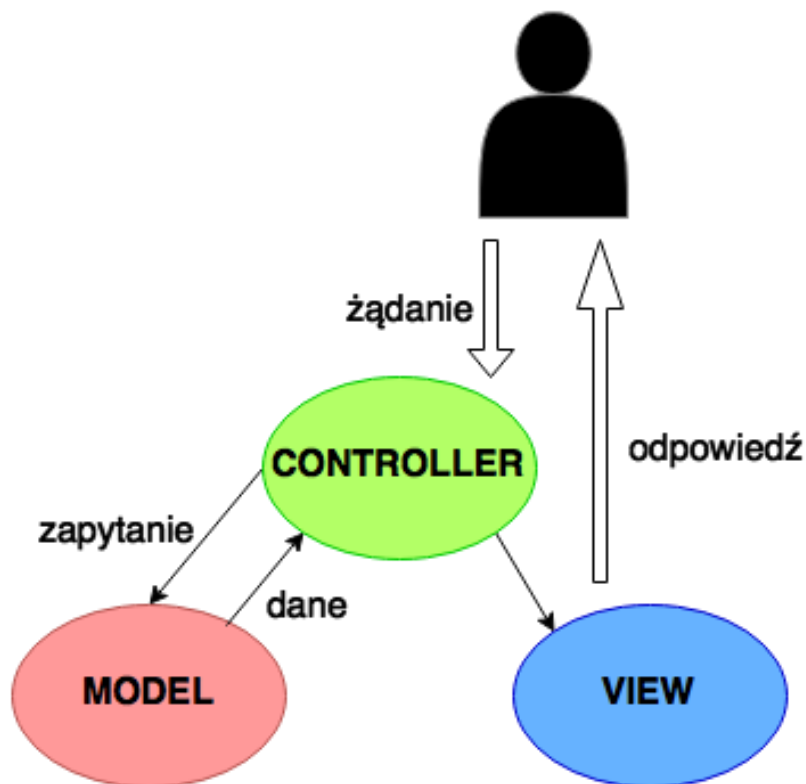
Back-end to inaczej część strony internetowej, będąca niewidzialna dla użytkownika, ale dostępna dla serwera, który przetwarza stronę. Można kolokwialnie powiedzieć, że jest to silnik strony www.

Bazowym językiem, w którym tworzony jest silnik systemu do zarządzania projektami informatycznymi jest PHP. Jest to język skryptowy, tak jak JavaScript, poszerzający funkcjonalność i możliwości stron internetowych. Jest bardzo prosty – nie ma czegoś takiego jak deklaracja zmiennych, a co za tym idzie deklaracji typów zmiennych. W odróżnieniu od języka JavaScript, który pracuje po stronie klienta, PHP pracuje po stronie serwera WWW. W związku z czym nadaje się do tworzenia złożonych aplikacji, jak fora internetowe, sklepy, w których należy zarządzać dużą ilością danych. Najpopularniejszym sposobem programowania w tym języku jest programowanie obiektowe, odwzorowujące rzeczywistość. Język ten jak najbardziej wspiera technologie obiektowe .

Jednak pisanie kodu od zera w PHP byłoby bardzo czasochłonne, a dostępne technologie pozwalają na przyspieszenie tych prac. Dlatego do tworzenia zintegrowanego systemu zarządzania projektem informatycznym wykorzystano bibliotekę PHP: Codeigniter. Dostarcza on biblioteki m.in. do obsługi formularzy, połączenia z bazami danych pozwalając skupić się tylko i wyłącznie na konkretach. Jest lekki, w porównaniu do pozostałych frameworków nie wymaga wielu zasobów. Dla użytkowników nieprogramujących aplikacji w PHP jest bardzo prosty i szybki do nauki. Pracuje z wykorzystaniem wzorca MVC (Model – View –Controller).

MVC jako wzorzec jest najpopularniejszy pod względem utrzymania porządku w kodzie, nadając mu przejrzystość, szybkość i łatwą rozbudowę. Idea stworzona przez Trygvego Reenskauga dzieli kod na część odpowiedzialną za przetwarzanie danych (Model) i na część służącą do wyświetlania wyników (View). A czym jest Controller? Kontroluje działania w zależności od decyzji użytkownika, jak i odświeża widok po zmianach w modelu.

Stosując MVC nie możemy już stosować jednego folderu, a w nim gromadzić wszystkie pliki PHP albo stosować jeden plik, w którym znajdują się bloki z kodem HTML i danymi z MySQL. Charakterystyka podziału na przestrzenie niemające ze sobą wiele wspólnego, sprawia, że aplikacja jest przejrzysta i jej szybkość działania rośnie.



*Rysunek 1 Schemat działania wzorca MVC*

Ostatnią technologią działającą po stronie serwera są bazy danych. W tym projekcie, do zarządzania bazami danych wykorzystano MySQL, jeden z najpopularniejszych systemów, którego walorami jest domyślny, prosty interfejs, elastyczność działania i szybkość działania. Baza danych MySQL łączy się z PHP, w tym przypadku z biblioteką Codeigniter, poprzez przekazanie do pliku PHP danych niezbędnych do komunikacji. Jest to wystarczający system do baz danych, ponieważ nie wykorzystujemy skomplikowanych zapytań, nie mamy rady również w MySQL stworzyć profesjonalnej hurtowni danych albo ich analizy. Gdyby zaistniała taka potrzeba, wtedy można skorzystać z płatnych narzędzi firmy Oracle.

### 3.5.3 Zintegrowane środowisko programistyczne

Aby to wszystko napisać, potrzeba wygodnego narzędzia do programowania, które pisząc kod jest w stanie podpowiedzieć znacznik, albo odnieść się do funkcji znajdującej się w innym pliku. Do tworzenia komponentów, kodu źródłowego, zapytań baz danych najwygodniej wykorzystać tak zwane zintegrowane środowisko programistyczne.

Wykorzystywanym przy pisaniu systemu, o którym mowa w temacie pracy inżynierskiej, był program PHPStorm firmy JetBrains. Jest on specjalnie przystosowany do pracy z aplikacjami pisanymi w języku PHP. Daje możliwość podpowiadania klas, metod, zmiennych, funkcji. W łatwy sposób można zmienić nazwę pliku lub klas, powodując zmianę nazwy w plikach, które odwołują się do nich. Przydatną funkcjonalnością jest możliwość debugowania kodu PHP jak i JavaScript. Również można bezpośrednio wysyłać zmienione pliki na serwer za pomocą protokołu FTP.

PHPStorm wspiera testy jednostkowe, systemy kontroli wersji, takie jak GIT, SVN, czy inne. Można w nim nie tylko pisać kody PHP, ale również nadaje się do pisania zwykłych stron HTML z wykorzystaniem kaskadowych arkuszy stylów. Jest lekkim programem, w porównaniu do swoich konkurentów. Jest szybki w działaniu, a design jest przyjazny dla długotrwałej pracy przed komputerem po ustawieniu ciemnego motywu.



## **4. Analiza wymagań systemu**

Rozdział ten zawiera zebrane i przeanalizowane wymagania względem aplikacji. Poprzez dokładną analizę wymagań określono charakterystykę i jakość systemu, tak aby był pożyteczny i wartościowy dla użytkownika. Wymagania zostały podzielone ze względu na swoją rolę w systemie na: wymagania funkcjonalne i wymagania niefunkcjonalne. Przedstawiono również wybrane scenariusze użytkownika, które przedstawiają zachowania aplikacji podczas możliwych sytuacji, w trakcie jej użytkowania.

### **4.1 Wymagania funkcjonalne**

Po zebraniu i przeanalizowaniu wymagań funkcjonalnych mamy możliwość zidentyfikowania i zaprezentowania zamierzonych zachowań systemu. Wymagania funkcjonalne przedstawiają funkcjonalności jakie system ma do zaoferowania użytkownikowi oraz określają jego zachowanie na zamierzone lub nie, sytuacje. Te wymagania opisują możliwości systemu w zakresie zachowania oraz dostępnych operacji.

#### **Dostęp do systemu**

Aby skorzystać z narzędzi jakie daje system do zarządzania projektem informatycznym, użytkownik powinien zalogować się do systemu przez formularz znajdujący się na stronie głównej. Jeśli nie posiada konta powinien zarejestrować się za pomocą formularza, do którego odniesienie znajduje się pod obszarem logowania na stronie głównej.

#### **Praca z projektami**

Każdy użytkownik ma trzy dostępne obszary pracy. Wykonywane, to projekty, do których został zaproszony jako pracownik. Obserwowane, to projekty dla których jest klientem. Stworzone, to projekty utworzone przez użytkownika, będące jego własnością.

#### **Wykonywanie projektów**

Praca nad projektami w formie programisty albo innego pracownika daje możliwość zobaczenia jakie zostały przydzielone mu zadania, które należy wykonać w danym dniu i które już zakończono w zależności od statusu.

### **Obserwowanie projektów**

Będąc klientem posiadamy możliwość obserwowania projektu, na jakim jest etapie produkcji. Klient ma prawo ocenić czy wykonana funkcjonalność zgadza się z jego założeniem czy nie. W przypadku negatywnej oceny zadanie wraca do programisty, który wcześniej się tym zajmował wraz ze wskazówkami od klienta.

### **Tworzenie projektów**

Każdy użytkownik aplikacji może stworzyć projekt, do którego zaprosi swoich ludzi, oraz klienta, jeśli tylko zechce. W każdej chwili może usunąć pracownika z projektu, jeśli będzie niepotrzebny. Manager określa zadania do wykonania nadając im termin i osoby, które je wykonają.

### **Edycja danych**

Na wypadek zmiany e-mail lub konieczności zmiany hasła, użytkownik zawsze może skorzystać z opcji edycji danych. Jest to niezbędny element każdej aplikacji, gdzie występuje możliwość tworzenia i zarządzania własnym kontem.

## **4.2 Wymagania нефunkcjonalne**

### **Użyteczność**

System do zarządzania projektami informatycznymi musi posiadać prosty w obsłudze interfejs. Wtedy nie będzie konieczności przeszkalania pracowników firmy, którzy pracują z tym systemem, co odbije się na oszczędnościach firmy. Wszelkie zaistniałe problemy przy użytkowaniu systemu będzie można w szybki sposób zgłosić osobie administrującej.

### **Niezawodność**

System może nie funkcjonować raz w miesiącu z powodu aktualizacji, którą administratorzy będą przeprowadzać. Jest to niezbędna procedura, w celu zapewnienia bezpieczeństwa i zachowania rozwoju aplikacji.

### **Wydajność**

Serwer we wzorcu MVC wymienia niewielkie paczki danych, co nie będzie obciążać systemu, gdy z niego będzie korzystała duża liczba użytkowników jednocześnie.

## **Bezpieczeństwo**

Wszelkie operacje bezpośrednio w bazie danych lub w kodzie aplikacji wykonuje tylko i wyłącznie administrator. Reszta użytkowników ma prawo korzystać z graficznej obsługi systemu specjalnie stworzonej do tych celów.

### **4.3 Scenariusze użytkownika**

Rozdział zawiera wybrane scenariusze użytkownika, które prezentują zachowania podczas konkretnych możliwych sytuacji podczas użytkowania aplikacji.

#### **4.3.1 Logowanie do systemu**

##### **Atrybuty:**

Główny aktor: Użytkownik

##### **Gwarancje powodzenia:**

Zalogowanie się i przekierowanie do panelu zarządzania projektami.

##### **Główny scenariusz:**

1. Użytkownik wchodzi na stronę główną
2. Wpisuje login w postaci adresu e-mail
3. Wpisuje hasło do konta
4. Potwierdzenie zgodności danych
5. Przekierowanie do panelu zarządzania projektami

##### **Rozszerzenia:**

1. Użytkownik podał zły login lub/i hasło
  - a) Wyświetlenie komunikatu o niepoprawnych danych logowania
  - b) Konieczność sprawdzenia loginu i hasła i dokonania poprawy

#### **4.3.2 Zmiana hasła**

##### **Atrybuty:**

Główny aktor: Użytkownik

**Gwarancje powodzenia:** Zapis nowego hasła do bazy danych

##### **Główny scenariusz:**

1. Użytkownik wchodzi do panelu ustawień
2. W przestrzeni zmiany hasła podaje aktualne i nowe wraz z jego powtórzeniem
3. Zatwierdzenie danych
4. Zaktualizowanie nowego hasła w bazie danych

##### **Rozszerzenia:**

1. Użytkownik podał aktualne złe hasło
  - a) Wyświetlenie komunikatu o niepoprawnym bieżącym hasle
  - b) Konieczność sprawdzenia starego hasła i dokonania poprawy
  - c) Zatwierdzenie danych

#### **4.3.3 Tworzenie nowego projektu**

##### **Atrybuty:**

Główny aktor: Użytkownik

**Gwarancje powodzenia:** Zarejestrowanie nowego projektu

##### **Główny scenariusz:**

1. Użytkownik wchodzi w zakładkę Stworzone w panelu zarządzania projektami
2. Klika w przycisk „Dodaj kolejny”
3. Pojawia się okno pop-up z formularzem
4. Podaje nazwę projektu, planowy termin jego rozpoczęcia i zakończenia
5. Zatwierdza dane
6. Nowy projekt zostaje zarejestrowany w bazie

**Rozszerzenia:**

1. Użytkownik nie uzupełnił pola nazwy lub daty
  - a) Wyświetlenie komunikatu o nieprawidłowym wypełnieniu formularza
  - b) Konieczność uzupełnienia wszystkich bloków przez użytkownika
2. Data końcowa projektu jest wcześniejsza niż data rozpoczęcia projektu
  - a) Wyświetlenie komunikatu o nieprawidłowym wypełnieniu formularza
  - b) Ustawienie prawidłowej daty początkowej i końcowej przez użytkownika

**4.3.4 Dodawanie nowego programisty****Atrybuty:**

Główny aktor: Użytkownik

**Gwarancje powodzenia:** Dodanie pracownika do projektu

**Główny scenariusz:**

1. Użytkownik wchodzi w zakładkę „Zaproś do projektu” w panelu projektu
2. Wpisuje adres e-mail osoby, którą chce zaprosić
  - a) Jeśli ta osoba istnieje w bazie, obramowanie bloku zmienia kolor na zielony
  - b) Jeśli osoba nie istnieje w bazie, obramowanie bloku zmienia kolor na czerwony
3. Zaprasza osobę poprzez kliknięcie przycisku „Dodaj”
4. Wysyłane jest powiadomienie na adres e-mail o zaproszeniu
5. Użytkownik potwierdza uruchamiając link dołączony w wiadomości e-mail
6. Użytkownik zostaje dodany do projektu

**Rozszerzenia:**

1. Użytkownik wprowadza niepoprawny składniowo adres e-mail
  - a) Wyświetlenie komunikatu o nieprawidłowym wypełnieniu formularza
  - b) Konieczność poprawienia adresu e-mail w bloku przez użytkownika

#### **4.3.5 Tworzenie zadania do wykonania**

##### **Atrybuty:**

Główny aktor: Użytkownik

**Gwarancje powodzenia:** Stworzenie i zapisanie zadań do wykonania dla danego projektu

##### **Główny scenariusz:**

1. Użytkownik wchodzi w zakładkę „Dodaj zadania” w panelu projektu
2. Za pierwszym uruchomieniem widać pierwszy pusty wiersz zadania
3. Wpisuje w pierwszym wierszu treść zadania określając daty i pracownika
4. Dodaje nowe zadanie klikając przycisk „Dodaj kolejny wiersz”
5. Uzupełnia kolejny wiersz wszystkimi danymi
6. Klika w przycisk zapisz wszystko
7. Wychodzi z panelu edycji zadań

##### **Rozszerzenia:**

1. Użytkownik nie zaznaczył żadnego wiersza podczas dodawania nowego
  - a) Wyświetlenie komunikatu o konieczności zaznaczenia wiersza
  - b) Dodanie kolejnego wiersza poprzez wywołanie zdarzenia po wciśnięciu przycisku
2. Użytkownik nie zapisał przed wyjściem swojego projektu
  - a) Zabezpieczenia aplikacji zapisują zdarzenia na bieżąco
  - b) Istnieje zagrożenie że tylko ostatnie zadanie nie zostało zapisane
  - c) Istnieje możliwość utraty ostatniego zadania

#### **4.3.6 Kontakt z administratorem**

##### **Atrybuty:**

Główny aktor: Użytkownik

**Gwarancje powodzenia:** Wysłanie zgłoszenia o nieprawidłowym działaniu systemu

**Główny scenariusz:**

1. Użytkownik uruchamia okno pop-up klikając na symbol koperty w menu
2. Uzupełnia formularz kontaktowy zgłaszając problem
3. Wysyła formularz klikając na przycisk „Zgłoś”
4. Czeka na odpowiedź
5. Otrzymuje odpowiedź zwrotną o rozwiązaniu problemu

**Rozszerzenia:**

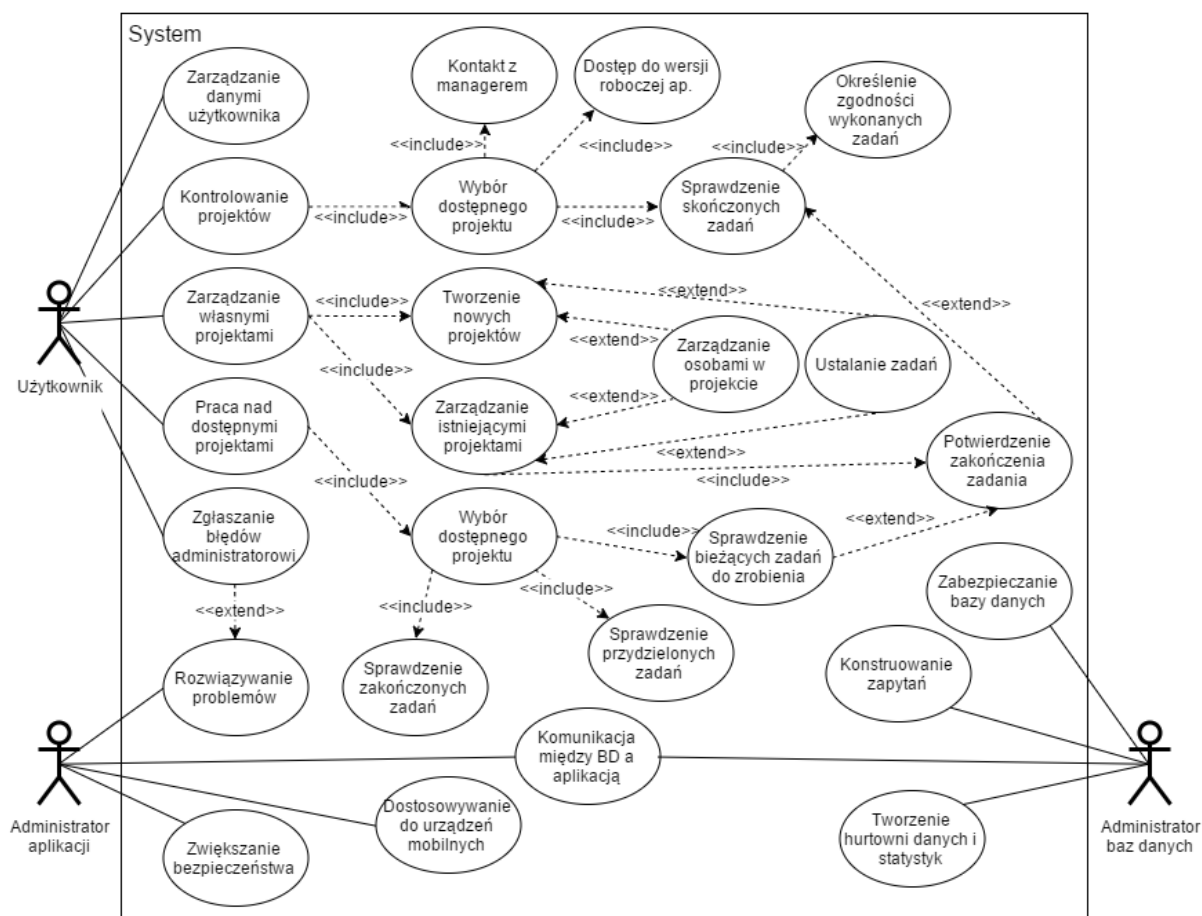
1. Użytkownik wprowadza niepoprawny składniowo adres e-mail
  - a) Wyświetlenie komunikatu o nieprawidłowym wypełnieniu formularza
  - b) Konieczność poprawienia adresu e-mail w bloku przez użytkownika

## 5. Modelowanie systemu

Unified Modeling Language to język znormalizowany, który pozwala zapisywać projekty systemu w różnej formie. Generalnie stosuje się go do obrazowania i specyfikowania, tworzenia i dokumentowania artefaktów zbudowanych podczas tworzenia systemu informatycznego lub aplikacji internetowej. UML w znaczny sposób ułatwia przepływ informacji pomiędzy osobami związanymi z jakimś konkretnym projektem (managerem, programistami, testerami, analitykami). Pomimo tego, że został nazwany językiem, to nie jest on językiem programowania, lecz bardziej językiem modelowania.

### 5.1 Modelowanie granic systemu

Pierwszy z diagramów określa funkcjonalność i otoczenie całego systemu. Jego własności przedstawiane są w postaci graficznej, w taki sposób, w jaki widziane są one od strony osoby korzystającej z systemu, czyli użytkownika. Można więc powiedzieć, że jest przedstawione wszystko to, co jest dostępne od strony front-endowej.



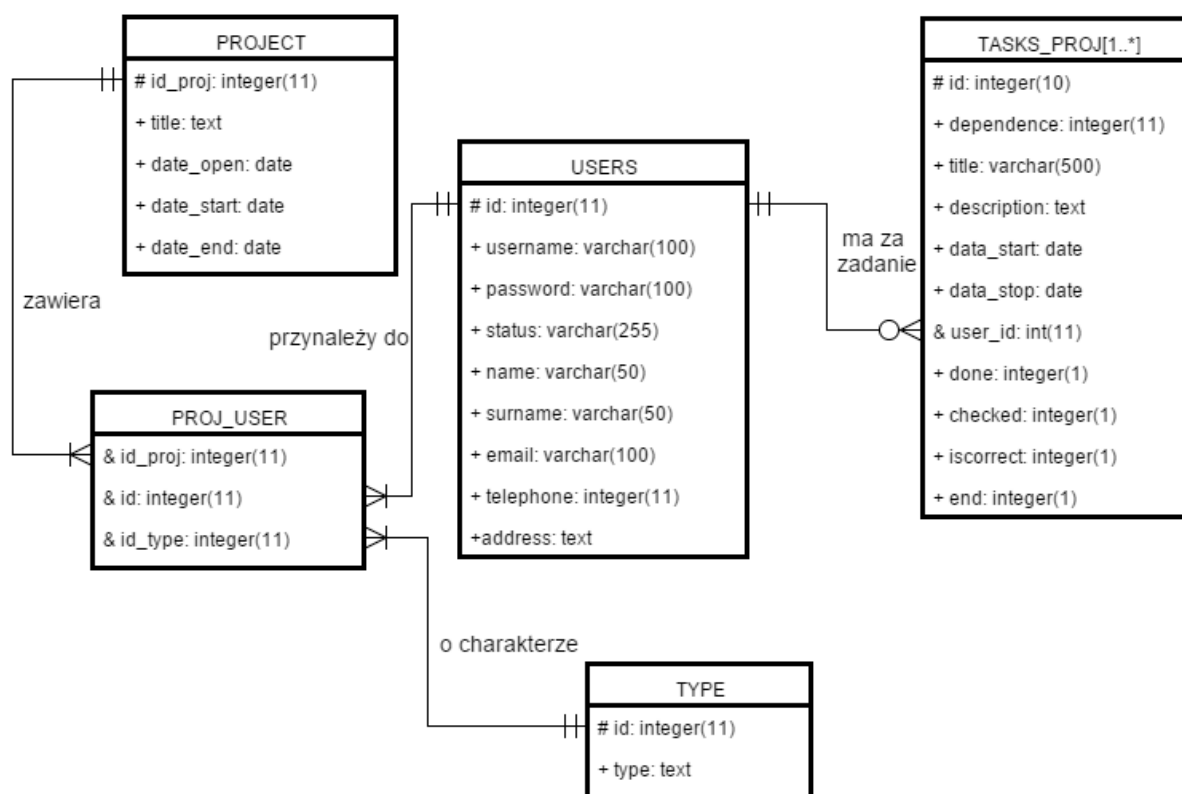
Rysunek 2 Diagram przypadków użycia



## 5.2 Modelowanie struktury baz danych

Diagram związków encji pozwala na graficzną prezentację encji i związków pomiędzy nimi aby ukazać budowę bazy danych. Dzięki takiemu wizualnemu zestawieniu można łatwo projektować i edytować strukturę, w celu jak najlepszej optymalizacji.

Każda encja posiada atrybuty, które mogą być różnych typów. Istnieje możliwość wystąpienia atrybutu, którego wartości są uniwersalne w danej encji, wtedy taki atrybut nazywany jest kluczem głównym. Jest to tak zwany identyfikator w bazie. Pomiedzy encjami można określić związek, tak zwaną relację. Relacja zależy od kluczy głównych i kluczy obcych. Możliwe stopnie relacji to: jeden do jeden, jeden do wielu, wiele do wielu.



Rysunek 3 Diagram związków encji

## 5.3 Modelowanie procesów

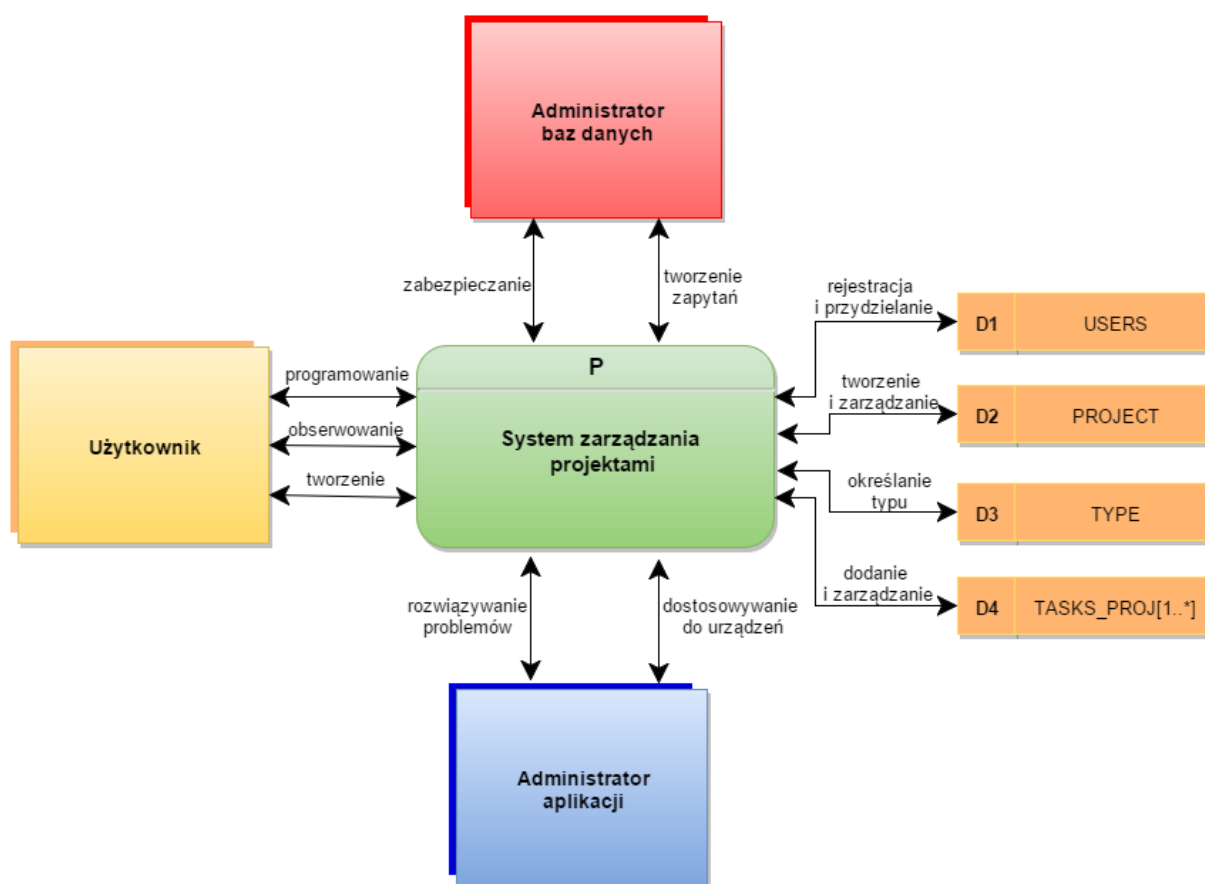
Inaczej zwany diagramem przepływu danych, prezentuje modele procesów w systemie. Stanowi on „mapę” procesów przedstawiających przepływ danych pomiędzy procesami w systemie, a także między środowiskiem a systemem. Buduje się go z czterech standardowych symboli, wykorzystując metodologię Gane-Sarsona:

- Procesy – to składniki systemu operujące na danych. Przesyłają i otrzymują dane za pośrednictwem ich przepływu
- Przepływy danych – to strumienie danych o zdefiniowanej zawartości, przemieszczające się pomiędzy dwoma obiektami (nadawcą i odbiorcą). Są to linie ze strzałkami określające kierunek przepływu danych.
- Obiekty zewnętrzne – to nadawcy, tacy jak użytkownicy, dostarczający informacji, które generują wykonywanie procesów w systemie.
- Magazyny danych – to punkty przechowywania danych pomiędzy procesami. Obiekt zewnętrzny nie ma bezpośredniego dostępu do magazynu. Tylko bezpośredni dostęp posiada proces.

W diagramach przepływu danych istnieje kilka reguł, o których należy pamiętać. Pierwszą z nich jest spostrzeżenie, że nie powinno być komunikacji pomiędzy obiektami zewnętrznymi jak i pomiędzy magazynami danych. Nie powinno być bezpośredniej komunikacji pomiędzy magazynami danych, a obiektami zewnętrznymi..

### 5.3.1 Modelowanie komunikacji systemu z obiektami

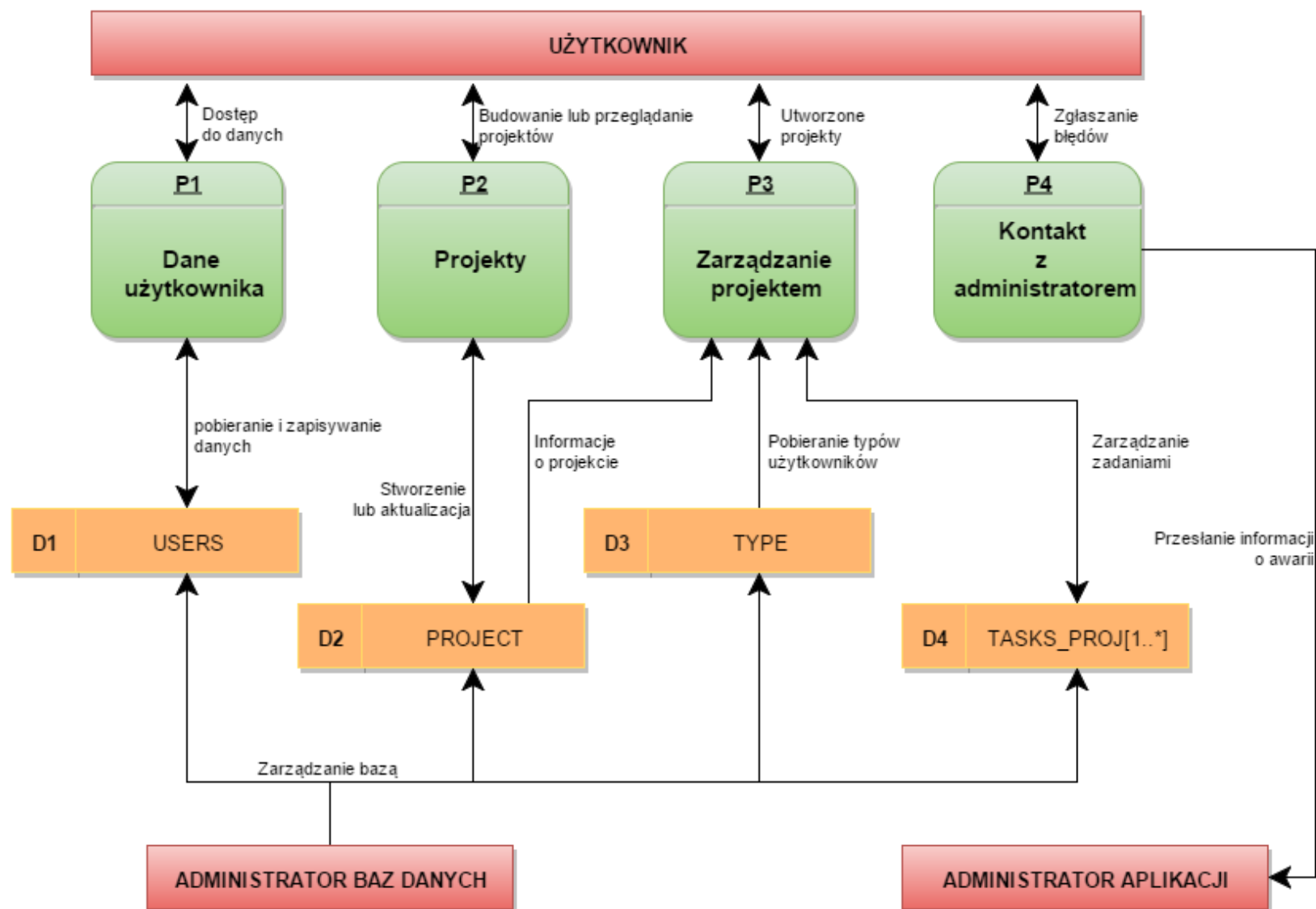
Pierwszy typ diagramu od którego zaczniemy to kontekstowy. Na samym początku konieczne jest określenie obiektów zewnętrznych – użytkowników. Na diagramie kontekstowym przedstawia się jeden reprezentacyjny proces tworzonego systemu. Ma on za zadanie pokazać proces komunikacji obiektów zewnętrznych z systemem.



Rysunek 4 Diagram DFD kontekstowy

### 5.3.2 Modelowanie głównych funkcjonalności

Po przedstawieniu jednego dużego procesu w diagramie kontekstowym, teraz rozbijemy go na mniejsze, również kluczowe, procesy pojawiające się w aplikacji. Każdy z tych procesów będzie odpowiedzialny za obsługę co najmniej jednego przepływu. W tym diagramie nie powinno się zapominać o obiektach zewnętrznych, czy magazynach danych, które pojawiły się już w diagramie kontekstowym. Dodatkowo, na poziomie aktualnego diagramu powstaje więcej dokładniejszych przepływów pomiędzy procesami i magazynami danych, oraz pomiędzy nowymi procesami, a obiektami zewnętrznymi, powstającymi na skutek rozpadu procesów z wcześniejszego diagramu.

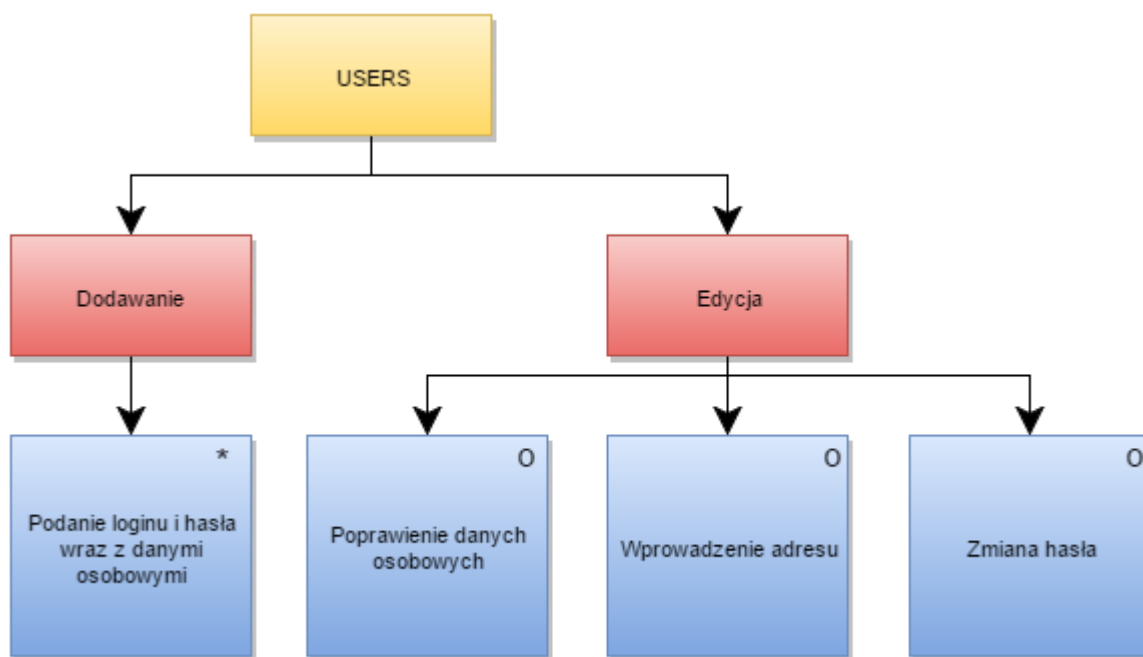


Rysunek 5 Diagram DFD systemow

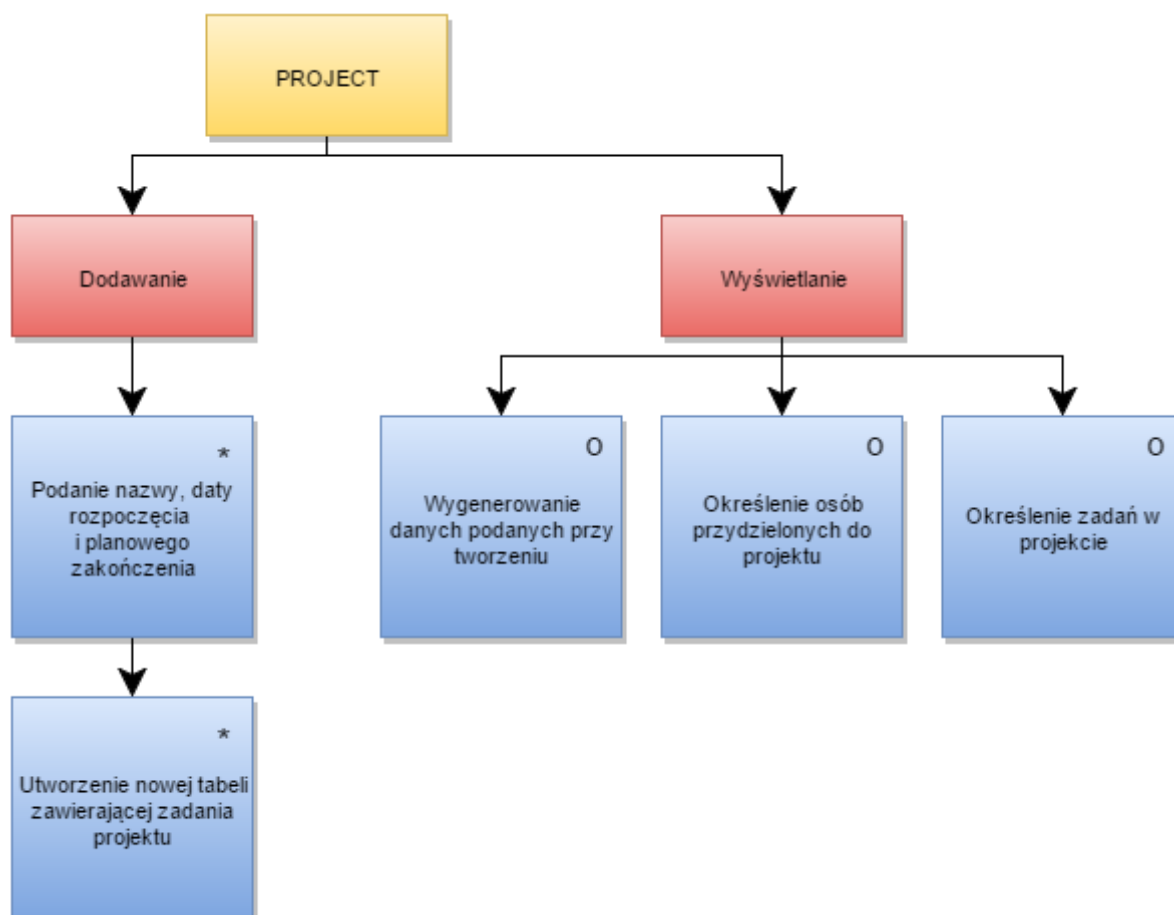
## 5.4 Modelowanie zdarzeń

Diagram historii życia encji ma strukturę drzewa, którego korzeniem jest konkretny obiekt, zaś węzłami są zdarzenia, które oddziałują na ten obiekt. Celem tego diagramu jest identyfikacja wszystkich zdarzeń oddziałujących na dany obiekt oraz sekwencji zdarzeń obiektu. Ważnym jest żeby miało to powiązanie z diagramem DFD. Należy przedstawić wszystkie błędne oraz wyjątkowe sytuacje, mające wpływ na obiekt.

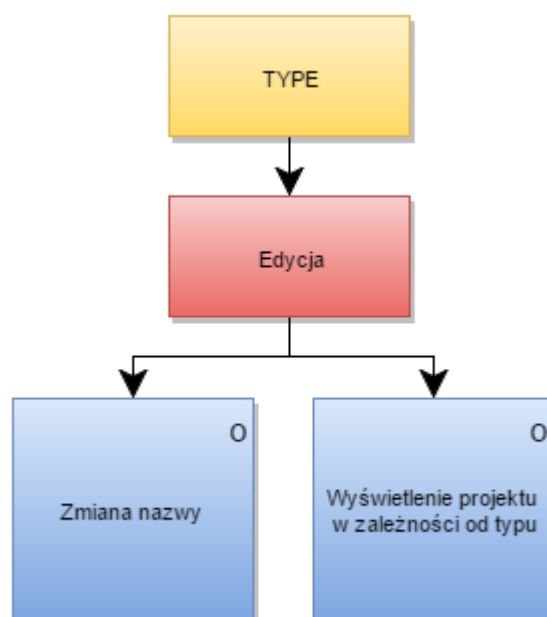
Diagram ELH ukazuje zdarzenia w postaci chronologicznej bez przedstawienia struktury procesów wywołanych podczas zaistnienia tych zdarzeń. Zdarzenia mogą być zewnętrzne, reprezentowane na DFD w postaci przepływu przecinającego granicę systemu. Mogą być też generowane zewnętrznie przez system. Mogą również być generowane przez czas.



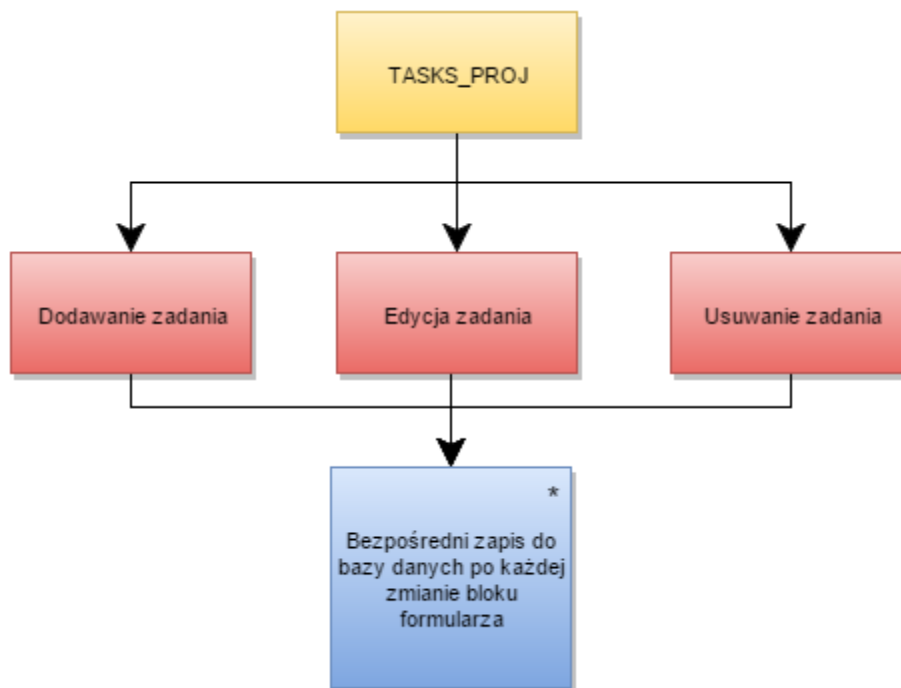
*Rysunek 6 Diagram ELH dla obiektu Users*



*Rysunek 7 Diagram ELH dla obiektu Project*



*Rysunek 8 Diagram ELH dla obiektu Type*



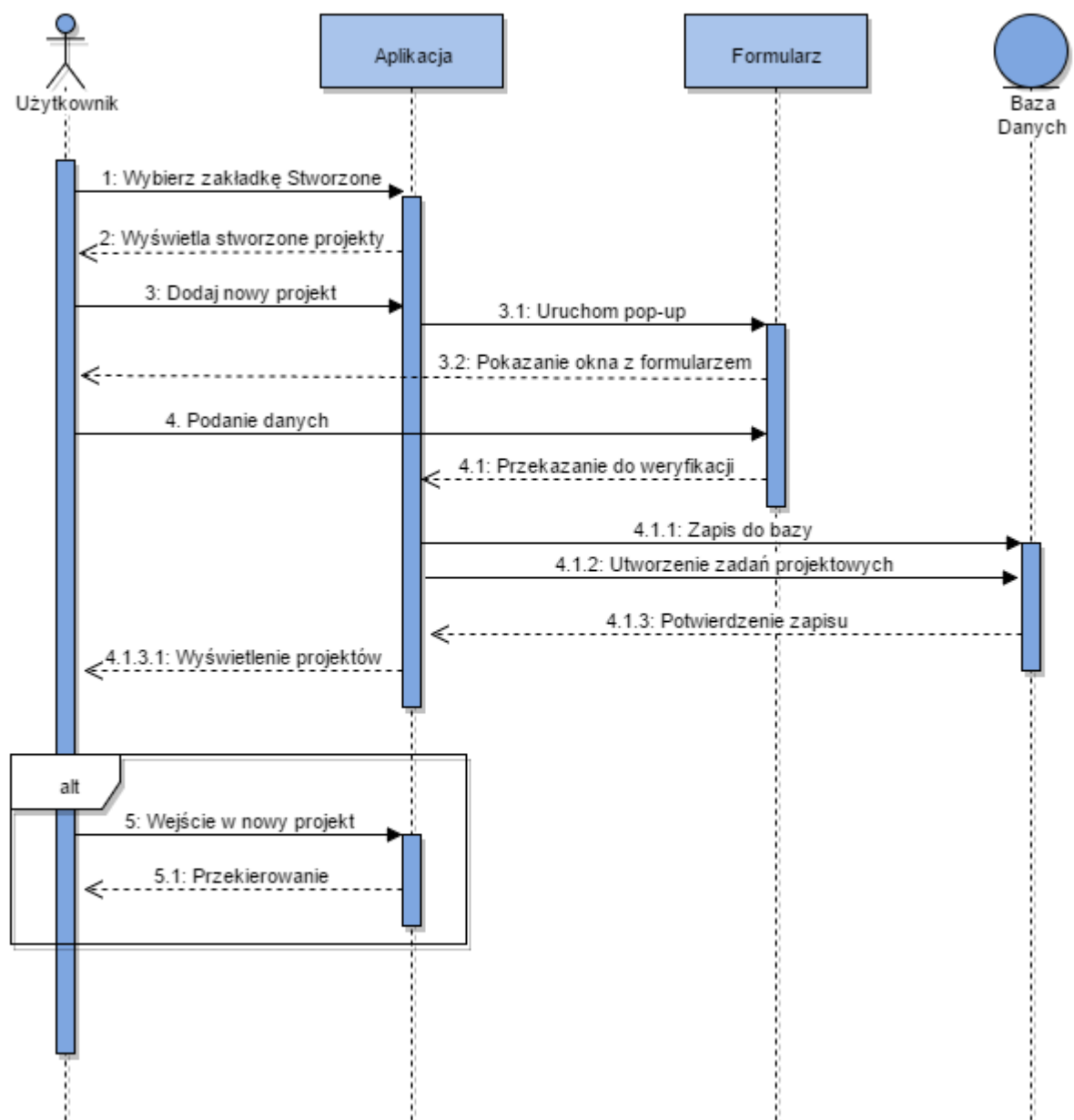
*Rysunek 9 Diagram ELH dla obiektu Tasks\_proj*

## 5.5 Modelowanie przepływu informacji

Diagramy tej grupy pozwalają na ukazanie połączenia obiektów między sobą wraz z przepływem informacji, który tam zachodzi. Schemat diagramu przypomina w pewien sposób układ współrzędnych. Obiekty ułożone są względem osi odciętych, natomiast komunikaty przesyłane są względem osi rzędnych. Kluczowym zastosowaniem diagramów sekwencji jest przedstawienie działania systemu, w zależności od interakcji użytkownika.

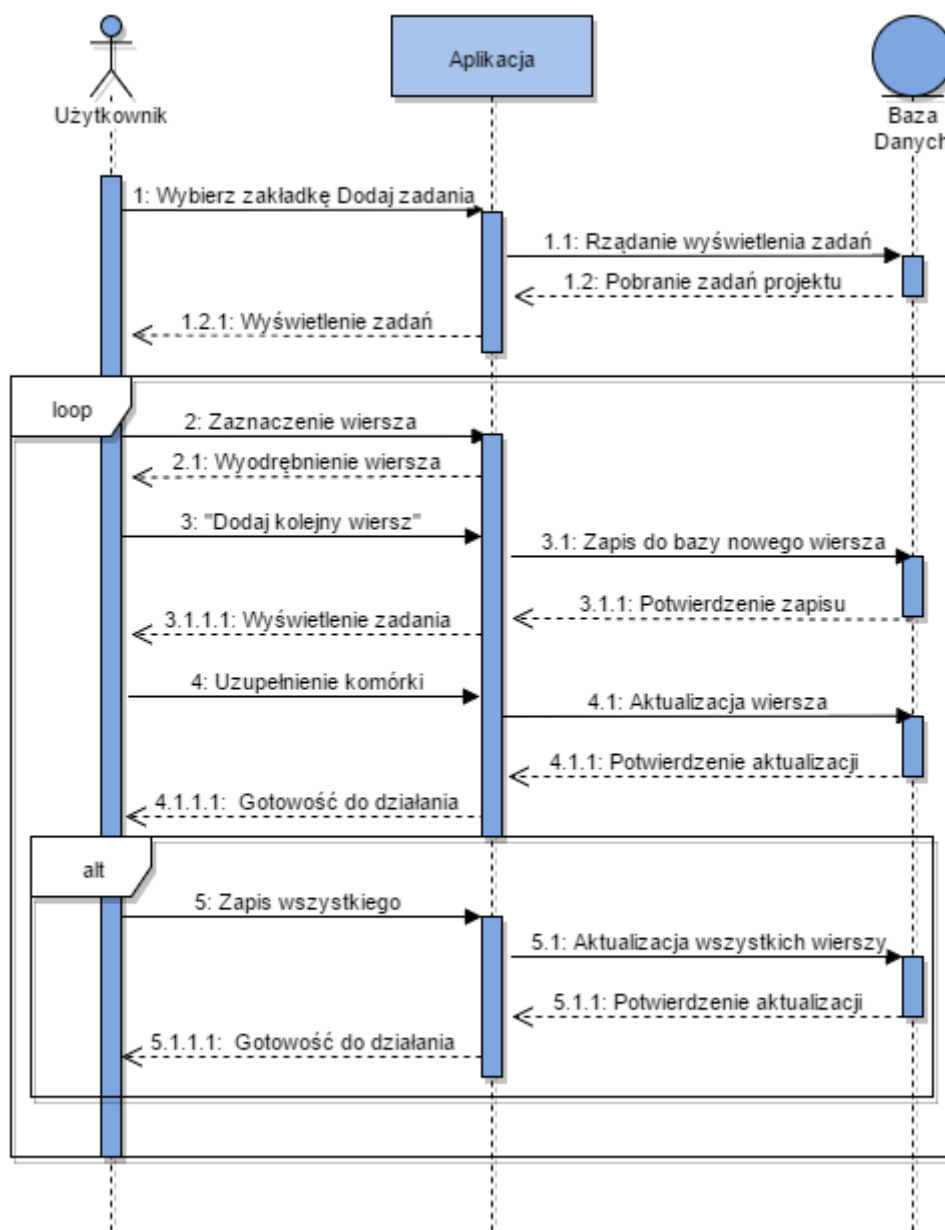
Od poprzednich diagramów różni się tym, że ma specyficzną notację. Występuje przede wszystkim linia życia określająca obiekt, bazę danych lub uczestnika interakcji i czas jego istnienia. Komunikat pozwala na przesyłanie informacji pomiędzy obiektami. Może on być synchroniczny, czekający na odpowiedź, albo asynchroniczny, który tej odpowiedzi nie planuje. Fragment pozwala rozszerzyć diagram o pętle, obszary warunkowe, przerwania.

## Tworzenie nowego projektu

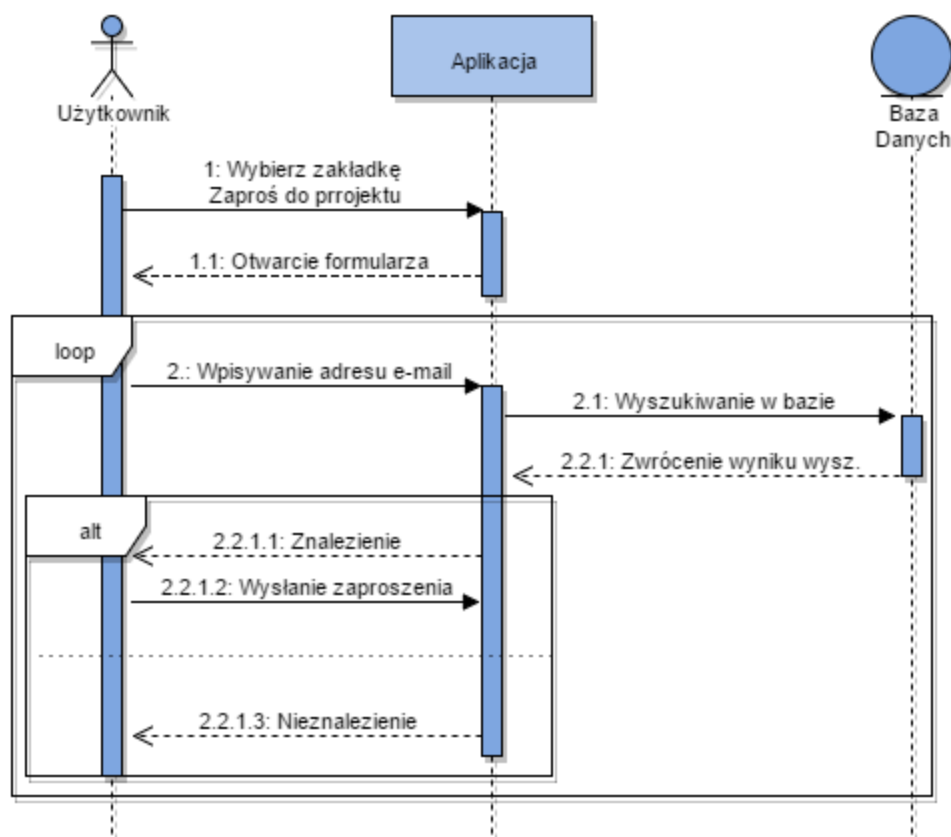




## Dodawanie nowych zadań do projektu



## Zaproszenie pracownika do współpracy przy projekcie



## 6. Projekt bazy danych

Zanim zaprojektowano bazę danych, należało przyjrzeć się problemowi zarządzania projektami informatycznymi. Trzeba było przemyśleć najlepsze i najbardziej optymalne rozwiązania, obserwując aplikacje innych firm, omówionych w rozdziale drugim. Po przeanalizowaniu wymagań funkcjonalnych projektu i problematyki związanej z zarządzaniem, postanowiono zaprojektować bazę danych, której schemat w postaci diagramu ERD jest widoczny w rozdziale piątym. Poniżej przedstawiono szczegółowy opis tabel.

Nazwa encji	Opis	Własności
PROJECT	Informacje na temat projektu, daty rozpoczęcia i zakończenia.	Każdy projekt ma swoją nazwę i terminy wykonania.
PROJ_USER	Informacje na temat przynależności osób do projektu.	Projekty mają osoby odpowiedzialne za zarządzanie lub tworzenie.
TYPE	Informacje dotyczące typów użytkowników.	Każdy użytkownik, należący do projektu ma swój typ użytkownika.
USERS	Informacje dotyczące użytkowników, danych logowania i personalnych.	Użytkownicy posiadają dane do logowania i dane personalne potrzebne do identyfikacji.
TASKS_PROJ [1..*]	Informacje dotyczące zadań, do konkretnego projektu w zależności od identyfikatora projektu.	Każdy projekt posiada zadania określone przez managera.

*Tabela 1 Określenie zbiorów encji*

Jeśli chodzi o relacje pomiędzy encjami i ich krotności to wyróżniamy jedno i tylko jedno wystąpienie encji (1..1), jedno lub wiele wystąpień (1..\*) oraz brak lub wiele wystąpień (0..\*).

PROJECT (1..1) → *posiada* → (1..1) TASKS\_PROJ

Każdy projekt posiada swój odpowiednik tabeli implementującej zadania

USERS (1..1) → *może pracować* → (1..\*) PROJ\_USER

Jeden użytkownik może pracować na co najmniej jednym projekcie jednocześnie

PROJECT (1..1) → *posiada* → (0..\*) TASKS\_PROJ

Jeden projekt posiada nieskończoną ilość zadań albo ani jednego zadania

## 6.1 Określenie atrybutów encji

Encja	Atrybuty	Typ danych (rozmiar)	Klucz główny	Klucz obcy	Wart. puste	Wart. domyśl.
PROJECT	id_proj	Integer(11)	Tak	Nie	Nie	Nie
	title	Text	Nie	Nie	Nie	Nie
	date_open	Date	Nie	Nie	Nie	Nie
	date_start	Date	Nie	Nie	Nie	Nie
	date_end	Date	Nie	Nie	Nie	Nie
PROJ_USER	id_proj	Integer(11)	Nie	Tak	Nie	Nie
	id	Integer(11)	Nie	Tak	Nie	Nie
	id_type	Integer(11)	Nie	Tak	Nie	Nie
TYPE	id_type	Integer(11)	Tak	Nie	Nie	Nie
	type	Text	Nie	Nie	Nie	Nie
USERS	id	Integer(11)	Tak	Nie	Nie	Nie
	username	Varchar(100)	Nie	Nie	Nie	Nie
	password	Varchar(100)	Nie	Nie	Nie	Nie
	status	Varchar(255)	Nie	Nie	Nie	Nie
	name	Varchar(50)	Nie	Nie	Nie	Nie
	surname	Varchar(50)	Nie	Nie	Nie	Nie
	email	Varchar(100)	Nie	Nie	Nie	Nie
	telephone	Integer(11)	Nie	Nie	Nie	Nie
TASKS_PROJ [1..*]	address	Text	Nie	Nie	Tak	null
	id	Integer(10)	Tak	Nie	Nie	Nie
	dependence	Integer(11)	Nie	Nie	Tak	null
	title	Varchar(500)	Nie	Nie	Tak	null
	description	Text	Nie	Nie	Tak	null
	data_start	Date	Nie	Nie	Tak	null
	data_stop	Date	Nie	Nie	Tak	null
	user_id	Integer(11)	Nie	Nie	Tak	null
	done	Tinyint(1)	Nie	Nie	Tak	null
	checked	Tinyint(1)	Nie	Nie	Tak	null
	isincorrect	Tinyint(1)	Nie	Nie	Tak	null
	correctdescript	text	Nie	Nie	Tak	null
	end	Tinyint(1)	Nie	Nie	Tak	null

*Tabela 2 Tabela atrybutów encji bazy danych*

Wyróżniono w encjach atrybuty, które są kluczami kandydującymi, tak samo jak klucze główne poprzez zagospodarowanie kolumn w tabeli. Do wyboru klucza głównego należało kierować się zasadą szybkości identyfikacji rekordów, uproszczenia tworzenia bazy i zwiększenia przejrzystości. Wartość klucza głównego powinna być odpowiednio dostosowana, by zapewnić unikalność dla każdego wystąpienia.

## 6.2 Ważniejsze polecenia języka SQL

### Wyszukiwanie projektów w zależności od typu użytkownika

```
SELECT *
FROM project
JOIN proj_user
ON project.id_proj = proj_user.id_proj
WHERE proj_user.id= $id_user
AND proj_user.id_type= $id_type;
```

### Zebranie dokładnych informacji o konkretnym projekcie

```
SELECT *
FROM project
JOIN proj_user
ON project.id_proj = proj_user.id_proj
JOIN users
ON users.id = proj_user.id
WHERE proj_user.id_proj= $id_project
AND proj_user.id = $id_user
AND proj_user.id_type= 1;
```

### Zestawienie użytkowników w zależności od typu

```
SELECT users.id AS `ajdi`, name, surname, email, type
FROM users
JOIN proj_user
ON users.id=proj_user.id
JOIN type
ON type.id_type = proj_user.id_type
WHERE proj_user.id_proj= $id_project;
```

### Sprawdzenie czy istnieje użytkownik o podanym id w konkretnym projekcie

```
SELECT id
FROM proj_user
WHERE proj_user.id_proj= $id_project
AND proj_user.id = $id_user;
```

## 6.3 Skrypty języka SQL

### Tworzenie tabeli PROJECT

```
CREATE TABLE IF NOT EXISTS `project` (  
  `id_proj` int(11) NOT NULL AUTO_INCREMENT,  
  `title` text NOT NULL,  
  `date_open` date NOT NULL,  
  `date_start` date NOT NULL,  
  `date_end` date NOT NULL,  
  PRIMARY KEY (`id_proj`)  
);
```

### Tworzenie tabeli PROJ\_USER

```
CREATE TABLE IF NOT EXISTS `proj_user` (  
  `id_proj` int(11) NOT NULL,  
  `id` int(11) NOT NULL,  
  `id_type` int(11) NOT NULL  
);
```

### Tworzenie tabeli TYPE

```
CREATE TABLE IF NOT EXISTS `type` (  
  `id_type` int(11) NOT NULL AUTO_INCREMENT,  
  `type` text NOT NULL,  
  PRIMARY KEY (`id_type`)  
);
```

### Tworzenie tabeli USERS

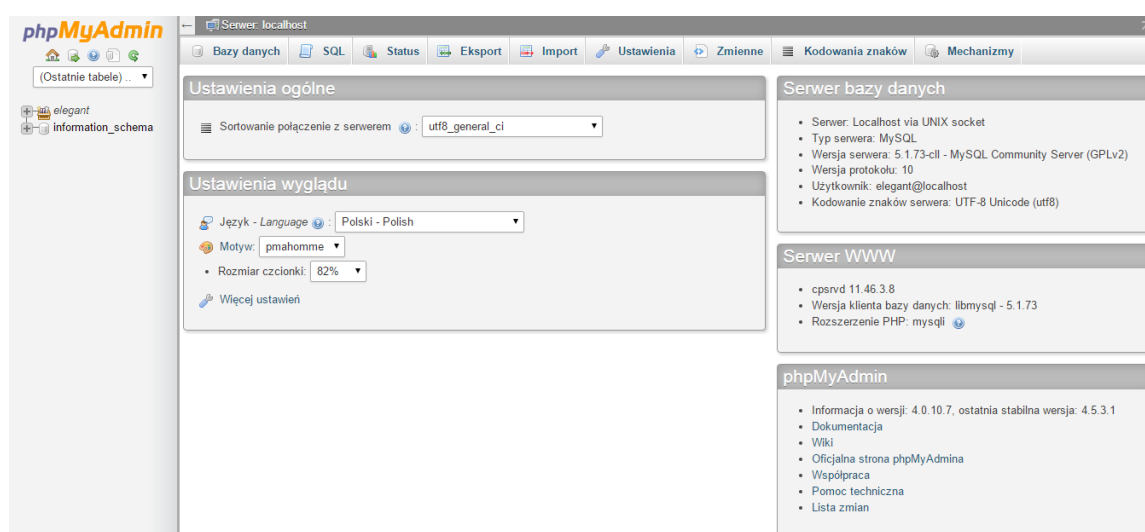
```
CREATE TABLE IF NOT EXISTS `users` (  
  `id` int(11) NOT NULL AUTO_INCREMENT,  
  `username` varchar(100) NOT NULL,  
  `password` varchar(100) NOT NULL,  
  `status` varchar(255) NOT NULL,  
  `name` varchar(50) NOT NULL,  
  `surname` varchar(50) NOT NULL,  
  `email` varchar(100) NOT NULL,  
  `telephone` int(11) NOT NULL,  
  `address` text,  
  PRIMARY KEY (`id`)  
);
```

### Tworzenie tabeli TASKS\_PROJ[1..\*]

```
CREATE TABLE IF NOT EXISTS `proj1` (  
  `id` int(10) unsigned NOT NULL AUTO_INCREMENT,  
  `dependence` int(11) DEFAULT NULL,  
  `title` varchar(500) DEFAULT NULL,  
  `description` text,  
  `data_start` date DEFAULT NULL,  
  `data_stop` date DEFAULT NULL,  
  `user_id` int(11) DEFAULT NULL,  
  `done` tinyint(1) DEFAULT NULL,  
  `checked` tinyint(1) DEFAULT NULL,  
  `incorrect` tinyint(1) DEFAULT NULL,  
  `correctdescript` text,  
  `END` tinyint(1) DEFAULT NULL,  
  PRIMARY KEY (`id`)  
);
```

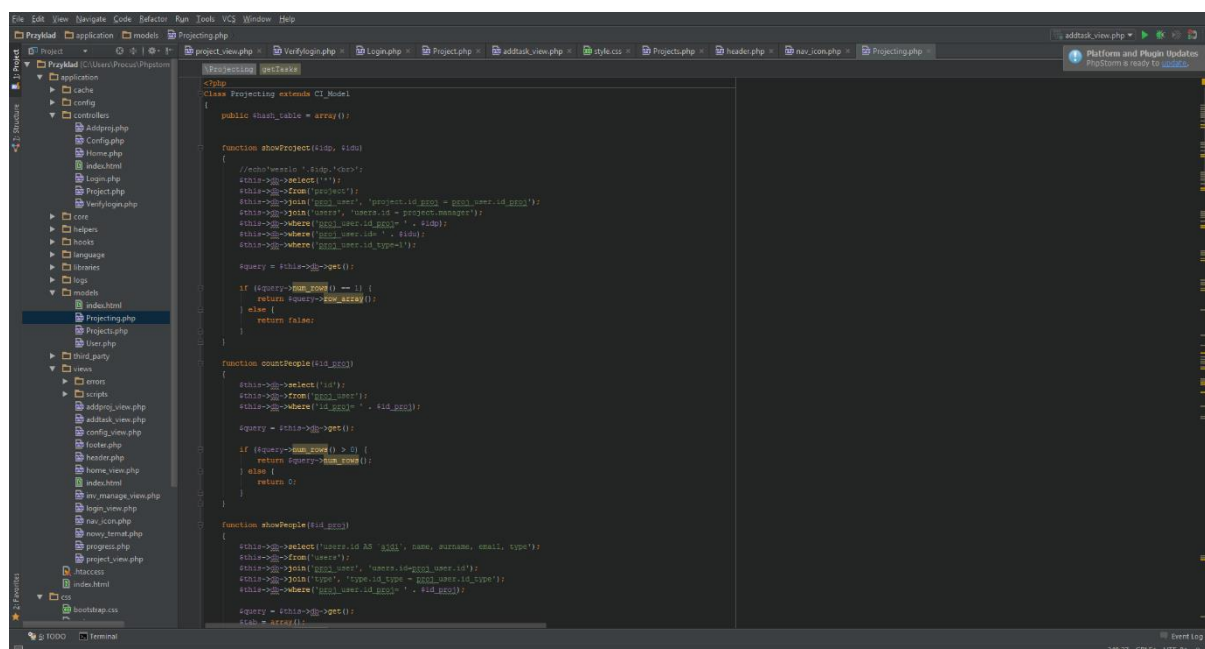
## 7. Dokumentacja techniczna dla administratora aplikacji i baz danych

Jak w każdym profesjonalnym systemie została dodana obsługa konkretnych zagadnień technicznych dla administratorów. Administrator baz danych do zarządzania bazą danych może wykorzystać wszelkie dostępne narzędzia służące do zarządzania bazą danych MySQL. Najpopularniejszym i sugerowanym programem jest phpMyAdmin, który w łatwy sposób pozwala na zarządzanie danymi. Nie jest tutaj wymagana zaawansowana umiejętność posługiwania się komendami do pracy nad bazami. Ważnym jest, aby serwer na którym będzie znajdowała się baza danych był serwerem bazodanowym MySQL.



**Rysunek 10 Panel powitalny narzędzia phpMyAdmin**

Aplikacja od strony administrowania serwisem nie posiada własnego interfejsu graficznego. Nie przewidziano specjalnego panelu zarządzania. Nie ma takiej potrzeby by stworzyć specjalne narzędzia do zarządzania samą stroną internetową systemu, ponieważ struktura jest zbyt prosta. Wszelkie poprawki, naprawy będą obejmowały kod źródłowy aplikacji. Aby zarządzać kodem potrzebne jest narzędzie, które pomimo swej prostoty użytkownika będzie potrafiło wspomóc administratora przy dokonywaniu edycji. Przy tworzeniu systemu, twórcy korzystali z narzędzia PhpStorm IDE firmy JetBrains - omówiono go w rozdziale trzecim. W związku z pracą nad kodem źródłowym wymagana jest od administratora dobra znajomość HTML, CSS, PHP i MySQL.



**Rysunek 11 Narzędzie PhpStorm do zarządzania kodem źródłowym systemu**

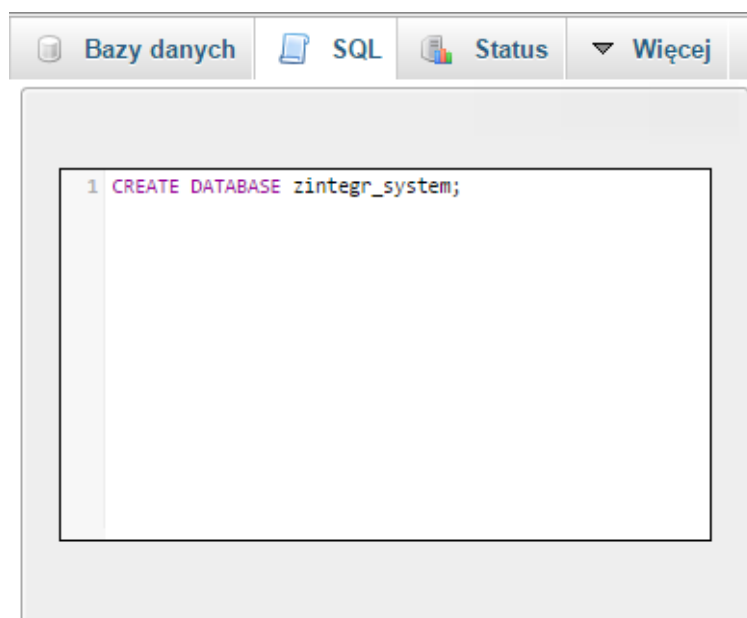
Pakiet instalacyjny, potrzebny do uruchomienia zintegrowanego systemu do zarządzania projektem informatycznym, to folder, zawierający w sobie pliki i foldery niezbędne do zainstalowania go na serwerze. Wszystkie pliki i foldery oprócz pliku *baza\_zrodlo.sql*, odnoszą się do aplikacji webowej, które trzeba przenieść na serwer.

application	2016-01-08 19:00	Folder plików	
css	2016-01-08 19:00	Folder plików	
img	2016-01-08 19:00	Folder plików	
js	2016-01-08 19:00	Folder plików	
system	2016-01-08 19:01	Folder plików	
user_guide	2016-01-08 19:04	Folder plików	
.ftpquota	2016-01-08 19:04	Plik FTPQUOTA	1 KB
	2016-01-08 19:04	Dokument tekstowy	1 KB
.htaccess	2016-01-08 19:04	Plik HTACCESS	1 KB
baza_zrodlo.sql	2016-01-08 19:08	Plik SQL	5 KB
composer.json	2016-01-08 19:04	Plik JSON	1 KB
contributing.md	2016-01-08 19:04	Plik MD	7 KB
index	2016-01-08 19:04	JetBrains PhpStorm	10 KB
license	2016-01-08 19:04	Dokument tekstowy	2 KB
readme.rst	2016-01-08 19:04	Plik RST	3 KB

**Rysunek 12 Pakiet zawierający pliki aplikacji oraz plik instalacyjny bazy danych**



Jednak zanim przetrzucimy pliki na serwer, najpierw należy przygotować bazę danych. W przykładzie wykorzystano narzędzie phpMyAdmin. Po zalogowaniu do niego, należy uruchomić zakładkę SQL, w której zostanie utworzona nowa baza danych pod system. Należy wpisać komendę



**Rysunek 13 Tworzenie nowej bazy danych pod table systemu**

Komenda, po wykonaniu polecenia utworzy nową bazę danych pod nazwą *zintegr\_system*. Teraz mając gotową bazę danych należy dodać do niej niezbędne tabele, które pozwolą na prawidłowe funkcjonowanie systemu. Należy przejść do bazy *zintegr\_system* i wybrać zakładkę *Import*. Z tego miejsca trzeba zaimportować plik SQL dostępny w pakiecie instalacji. Należy kliknąć przycisk *Wybierz plik*, wybrać w którym miejscu na komputerze znajduje się pakiet instalacyjny i wyodrębnić plik *baza\_zrodlo.sql*. Najlepiej ustawić kodowanie bazy danych na UTF-8. Po wybraniu należy wykonać import.

#### **Plik do importu:**

Plik może być skompresowany (gzip, bzip2, zip) bądź nie.

Plik skompresowany musi mieć rozszerzenie **[format].[kompresja]**, np. **.sql.zip**

Wyszukaj w komputerze:  baza\_zrodlo.sql (Maksymalny rozmiar: 50MB)

Kodowanie znaków pliku:

**Rysunek 14 Importowanie pliku konfiguracyjnego bazy danych**

Można zauważyć utworzone tabele na liście znajdującej się pod utworzoną bazą danych. Tabele omawiane były w rozdziale piątym, dotyczącym baz danych. Również w tym rozdziale zostały przedstawione komendy tworzenia poszczególnej tabeli. Wyjątek stanowią tabele TASKS\_PROJ. Powstają dopiero przy tworzeniu każdego nowego projektu. Numery każdej tabeli są uzależnione od numerów projektów, stworzonych przez osobę zarządzającą.



Rysunek 15 Struktura tabel po zaimportowaniu pliku SQL

Jeżeli wszystko do tej pory przebiegło pomyślnie, można teraz przejść do instalacji aplikacji na serwerze, której proces jest banalnie prosty. Na początku należy się zalogować na swój serwer internetowy. Dane do logowania są uzależnione od dostawcy usług, a instrukcja i dane do logowania, powinny być przez niego dostarczone. Do logowania najlepiej wykorzystać programy, takie jak Total Commander albo FileZilla, obsługujące przesyłanie plików za pomocą protokołu FTP lub innego, w zależności od serwera. Należy teraz przesłać wszystkie pliki z pakietu instalacji (oprócz *baza\_zrodlo.sql*) znajdującego się na komputerze, na serwer. Aby system był widoczny na poziomie roota, należy go przesłać do katalogu głównego na serwerze. Natomiast, jeśli miałby działać w podkatalogu roota, to należy przesłać pliki do tego podkatalogu.

↑ Nazwa	Roz.	Wielkość	Czas	Atryb	↑ Nazwa	Roz.	Wielkość	Czas	Atryb
[.]	<DIR>				[.]	<DIR>		2016-01-09 08:35	—
[application]	<DIR>		2015-11-13 16:34	-755	[application]	<DIR>		2016-01-08 19:00	—
[css]	<DIR>		2015-12-21 11:21	-755	[css]	<DIR>		2016-01-08 19:00	—
[img]	<DIR>		2015-12-21 14:13	-755	[img]	<DIR>		2016-01-08 19:00	—
[js]	<DIR>		2015-12-21 11:20	-755	[js]	<DIR>		2016-01-08 19:00	—
[system]	<DIR>		2015-11-13 16:37	-755	[system]	<DIR>		2016-01-08 19:01	—
[user_guide]	<DIR>		2015-11-13 16:42	-755	[user_guide]	<DIR>		2016-01-08 19:04	—
ftpquota		13	2015-12-14 00:37	-600	ftpquota		13	2016-01-08 19:04	-a-
.gitignore		489	2015-10-31 10:37	-644	.gitignore		489	2016-01-08 19:04	-a-
.htaccess		615	2015-11-13 17:03	-644	.htaccess		615	2016-01-08 19:04	-a-
composer	json	469	2015-10-31 10:37	-644	baza_zrodlo	sql	5 089	2016-01-08 19:08	-a-
contributing	md	6 502	2015-10-31 10:37	-644	composer	json	469	2016-01-08 19:04	-a-
index	php	9 805	2015-10-31 10:37	-644	contributing	md	6 502	2016-01-08 19:04	-a-
license	txt	1 114	2015-10-31 10:37	-644	index	php	9 805	2016-01-08 19:04	-a-
readme	rst	2 338	2015-10-31 10:37	-644	license	txt	1 114	2016-01-08 19:04	-a-
					readme	rst	2 338	2016-01-08 19:04	-a-

Rysunek 16 Po prawej: pakiet instalacyjny na komputerze; po lewej: pakiet instalacyjny na serwerze

Kolejnym krokiem jest konfiguracja systemu. Aby to zrobić należy przejść do folderu *application/config* i otworzyć plik *database.php*. Na samym dole pliku, po przeczytaniu instrukcji, znajdują się pola do uzupełnienia. Najważniejsze z nich są `hostname` (nazwa hosta – domyślnie `localhost`), `username` (nazwa użytkownika logującego się do bazy), `password` (hasło użytkownika logującego się do bazy) i `database` (nazwa bazy danych).

```
$db['default'] = array(
    'dsn' => '',
    'hostname' => 'localhost',
    'username' => 'elegant_piotr',
    'password' => '*****',
    'database' => 'elegant_inz',
    'dbdriver' => 'mysqli',
    'dbprefix' => '',
    'pconnect' => FALSE,
    'db_debug' => (ENVIRONMENT !== 'production'),
    'cache_on' => FALSE,
    'cachedir' => '',
    'char_set' => 'utf8',
    'dbcollat' => 'utf8_general_ci',
    'swap_pre' => '',
    'encrypt' => FALSE,
    'compress' => FALSE,
    'stricton' => FALSE,
    'failover' => array(),
    'save_queries' => TRUE
);
```

Skonfigurowanie bazy danych to niestety nie jest wszystko. Należy jeszcze ustawić główne katalogi systemu i główny plik wykonywalny w systemie. Aby to zrobić należy uruchomić plik *config.php* w folderze *config*. Pierwszą zmienną jaką należy określić to `$config['base_url']`. Mówi ona systemowi, jaki jest podstawowy adres URL aplikacji. Jeśli wrzucono plik instalacyjny do poziomu root, to zmienna będzie miała charakter: `$config['base_url'] = 'http://testowa.com/';` jeśli zostały stworzone jakieś podfoldery, to zmienna będzie miała postać: `$config['base_url'] = 'http://testowa.com/inz/podfolder/';`

Ostatnią opcjonalną rzeczą do ustawienia jest zmienna `$config['index_page'] = ''`; określająca ścieżkę do pliku startowego, inną niż *index.php*. Może tak się zdarzyć, ale w stworzonym systemie nie ma takiej potrzeby. Jeśli administrator systemu będzie potrzebował ustawić inny plik domyślny witryny, będzie mógł go tam wpisać. Domyślnie wartość zmiennej jest pusta, ponieważ korzysta z *index.php*.

Administrator powinien ustawić również w kodzie aplikacji swój adres e-mail, na który mają przychodzić wiadomości od użytkowników.

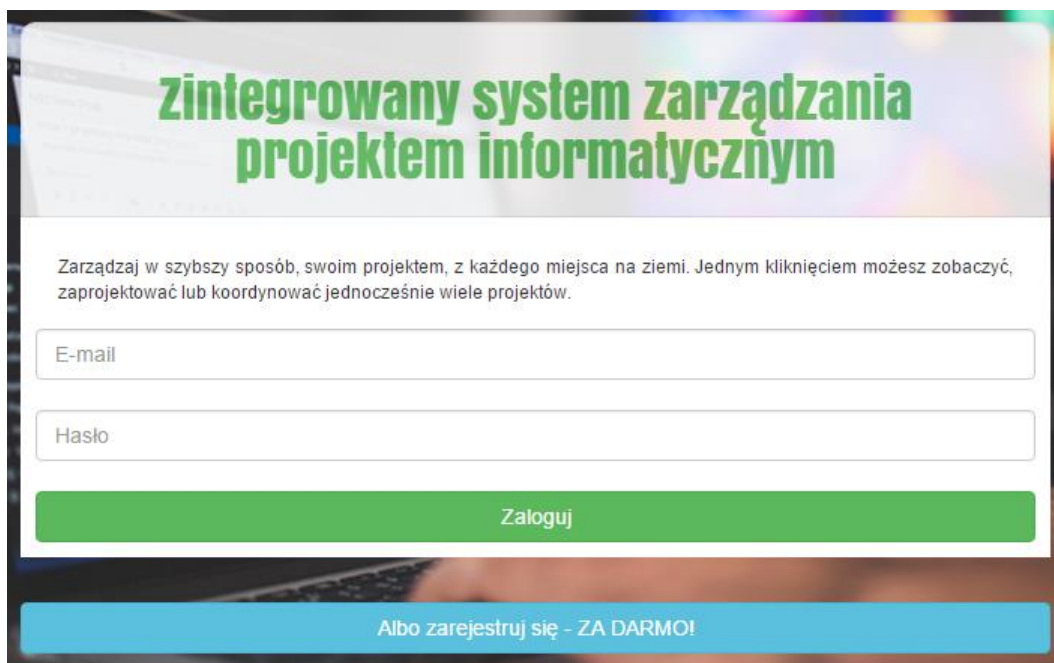
## 8. Dokumentacja techniczna dla użytkowników

Celem pracy było stworzenie aplikacji internetowej pozwalającej każdemu użytkownikowi niezależnie od wieku, korzystać w sposób łatwy i intuicyjny.

### 8.1 Logowanie do systemu

Po wpisaniu strony URL aplikacji w pasku adresu i załadowaniu przez serwer ukazuje się panel powitalny, na którym bezpośrednio można zalogować się do systemu. Aby zalogować się, należy mieć utworzone konto. Danymi do logowania jest adres e-mail i hasło podane podczas rejestracji.

Gdyby użytkownik zapomniał swojego hasła, istnieje możliwość przypomnienia go. Aby móc to zrobić, użytkownik powinien co najmniej raz podać niepoprawne hasło. Wyświetli się wtedy komunikat o nieprawidłowych danych z odnośnikiem pozwalającym na przypomnienie. Jeśli istnieje taki adres e-mail w bazie, to system wygeneruje nowe hasło i prześle go na podany wcześniej adres. W każdej chwili po zalogowaniu, użytkownik będzie mógł go zmienić na inny w panelu ustawień.

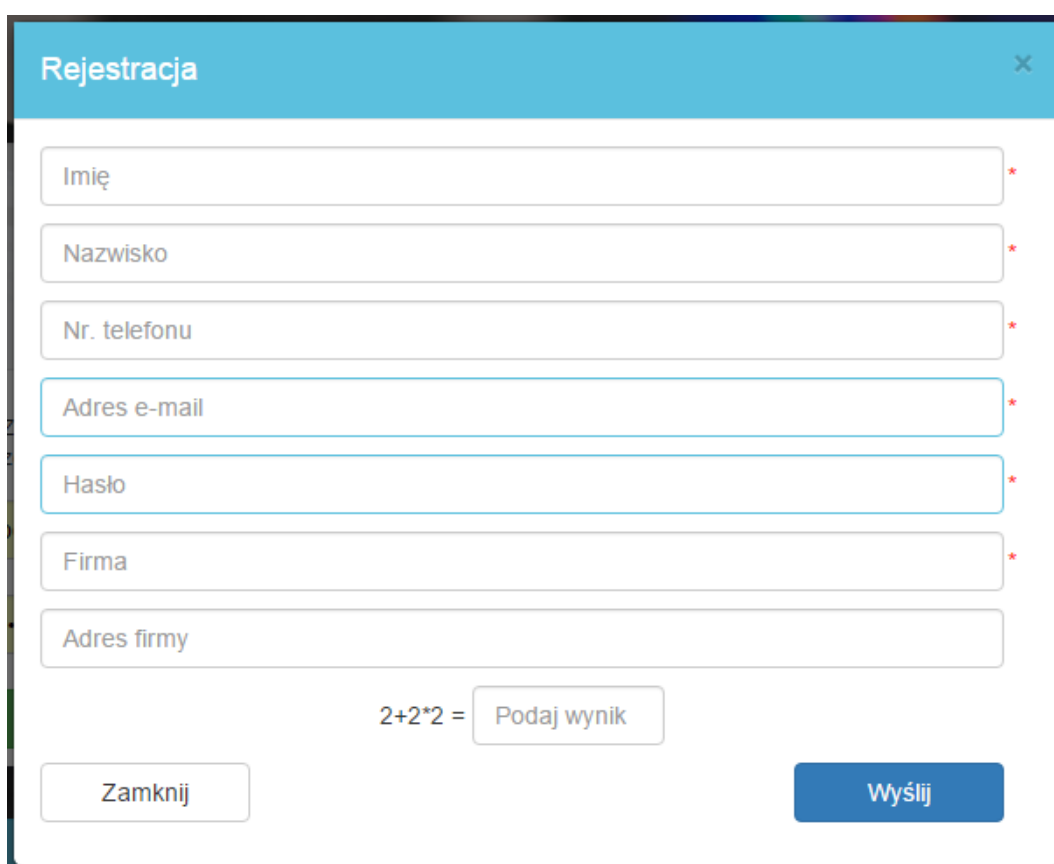


The image shows a login interface for a project management system. At the top, there is a header with the text "Zintegrowany system zarządzania projektem informatycznym" in green. Below the header, there is a paragraph of text: "Zarządzaj w szybszy sposób, swoim projektem, z każdego miejsca na ziemi. Jednym kliknięciem możesz zobaczyć, zaprojektować lub koordynować jednocześnie wiele projektów." Below this text, there are two input fields: "E-mail" and "Hasło". Below the input fields, there is a green button labeled "Zaloguj". At the bottom, there is a blue button labeled "Albo zarejestruj się - ZA DARMO!".

*Rysunek 17 Panel logowania do systemu zarządzania projektami*

## 8.2 Rejestracja do systemu

Chcąc mieć dostęp do funkcjonalności systemu, do zarządzania projektami informatycznymi użytkownik nie mający konta powinien zarejestrować się. Może tego dokonać wybierając na stronie startowej aplikacji przycisk mówiący o rejestracji. Pojawia się okno z formularzem, zawierającym podstawowe dane użytkownika. Wymagane pola oznaczone są gwiazdką (\*). Po uzupełnieniu pól, należy wysłać dane za pomocą przycisku. Jeśli wszystkie dane zostały poprawnie wprowadzone, system doda użytkownika do bazy danych i od tego momentu będzie mógł się już zalogować.



The image shows a registration window titled "Rejestracja" with a close button (X) in the top right corner. The form contains the following fields, each with a red asterisk (\*) indicating it is required:

- Imię
- Nazwisko
- Nr. telefonu
- Adres e-mail
- Hasło
- Firma
- Adres firmy

Below the fields is a CAPTCHA section with the text "2+2\*2 =" and a button labeled "Podaj wynik". At the bottom of the form are two buttons: "Zamknij" (Close) on the left and "Wyślij" (Send) on the right.

*Rysunek 18 Panel rejestracyjny dla użytkowników*

Ciekawym rozwiązaniem jest uniknięcie potwierdzenia założenia konta za pomocą linku aktywacyjnego wysyłanego na adres e-mail podany przy zakładaniu konta. Zamiast tak długiej drogi autoryzacji zastosowano technologie CAPTCHA, która weryfikuje czy użytkownik jest człowiekiem, czy robotem internetowym.

### 8.3 Panel użytkownika

System daje do wyboru różne tryby pracy przy projekcie. Można stworzyć projekt i nim zarządzać, można być osobą, która pracuje nad jego tworzeniem oraz można go obserwować jako klient. To wszystko znajduje się w 3 zakładkach dostępnych po zalogowaniu do systemu.

Wykonywane

Obserwowane

Stworzone

Nazwa	Dzień Utworzenia	Dzień rozpoczęcia	Dzień zakończenia
Badania naukowe 2016 IT	2015-11-16	2015-11-16	2015-11-30
Badania Naukowe 2015 IT	2015-11-16	2015-11-16	2015-11-30
Projekt zagraniczny	2015-11-16	2015-11-17	2015-11-20
Projekt pierwszy "Testowy"	2015-11-16	2015-11-17	2015-11-30

DODAJ KOLEJNY

*Rysunek 19 Panel zarządzający utworzonymi projektami*

Żeby wejść w szczegóły projektu wystarczy kliknąć w wiersz, w którym on się znajduje. Dotyczy to wszystkich opcji pracy nad projektami. System przekieruje użytkownika do szczegółowego panelu zarządzania projektem.

### 8.4 Zmiana danych osobowych lub hasła

Oprócz dostępu do projektów, użytkownik ma również możliwość edycji danych osobowych podawanych przy rejestracji. A także przewidziana jest możliwość zmiany hasła, gdyby użytkownik miał taką potrzebę. By dostać się do ustawień wystarczy kliknąć w przycisk ⚙️ (koło zębate), znajdującego się w górnych obszarach aplikacji.

The image shows two side-by-side web panels. The left panel, titled 'Zmień dane' (Change data), contains six text input fields with red asterisks indicating required fields. The fields contain the following text: 'Piotrrr', 'Tomaszewskibb', 'bob@o2.pl', '889555962', 'Programista Java firmy Motorola', and '11111123444'. A green 'Zmień' (Change) button is at the bottom. The right panel, titled 'Zmień hasło' (Change password), contains three text input fields with red asterisks. The first is 'Aktualne hasło' (Current password), the second is 'Nowe hasło' (New password), and the third is 'Powtórz nowe hasło' (Repeat new password). A green 'Zmień' (Change) button is at the bottom. Both panels are part of a larger interface with navigation icons at the top right.

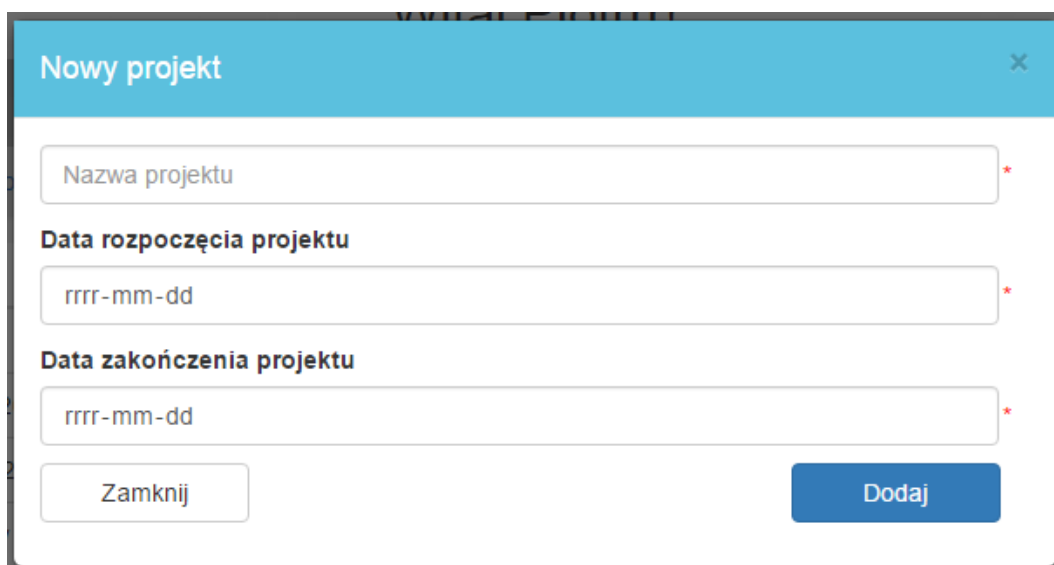
**Rysunek 20 Panel zmiany danych i hasła**

Użytkownik ma do wyboru zmianę swojego imienia, nazwiska, adresu e-mail, telefonu i adresu zamieszkania, przy czym adres zamieszkania nie jest wymagany. Po dokonaniu wszystkich zmian należy je zatwierdzić przyciskiem znajdującym się pod formularzem. Teraz można kontynuować pracę nad projektami.

Zmiana hasła jest równie prosta jak zmiana danych osobowych. Wystarczy znać aktualnie wykorzystywane hasło. Jest to zabezpieczenie przed niepożądanym zawłaszczeniem konta przez osobę trzecią, zupełnie niezwiązaną z systemem. Aby dokonać edycji istniejącego hasła wystarczy wprowadzić dane w polach do tego przeznaczonych. Jeśli aktualne hasło będzie niepoprawne, albo nowe hasło nie będzie zgodne z ponownie wprowadzonym nowym hasłem, to aplikacja zwróci komunikat o zaistniałym zdarzeniu. W przeciwnym wypadku hasło zostanie zaktualizowane i kolejne logowanie będzie odbywało się przy użyciu nowego hasła.

## 8.5 Tworzenie projektu

W zakładce „Stworzone” będą znajdować się projekty, które zostaną stworzone przez zalogowanego użytkownika. Aby zbudować taki projekt, wystarczy uruchomić okno klikając na przycisk „Dodaj kolejny” i uzupełnić pola zawierające nazwę projektu, datę rozpoczęcia i planowego zakończenia projektu. Z datą zakończenia projektu jest tak, że jeśli zadania wskażą, iż data zakończenia nie zgadza się z datą najpóźniejszego zadania, to zostanie wyświetlony komunikat o konieczności zaktualizowania daty zakończenia. System sam zaktualizuje tę datę jeśli manager na to pozwoli.



Nowy projekt

Nazwa projektu \*

Data rozpoczęcia projektu \*

Data zakończenia projektu \*

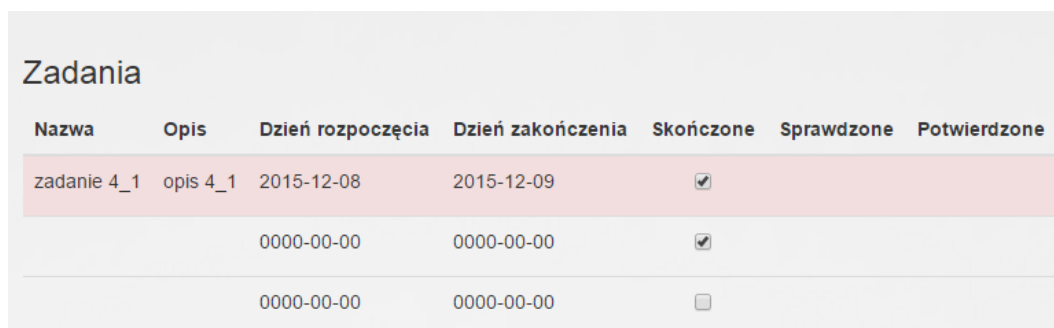
Zamknij Dodaj

Rysunek 21 Okno z formularzem do tworzenia nowego projektu

## 8.6 Praca nad projektem

W kolejnej zakładce „Wykonywane” znajdują się projekty do których zostaliśmy przydzieleni. Jeśli tak się nie stało, to wyświetlany jest komunikat o braku przydzielonych projektów. W przeciwnym razie mamy listę, z której wybiera się interesujący nas projekt.

Po wejściu w szczegóły projektu, aplikacja wyświetla szczegółowe informacje o projekcie i osobie, która go utworzyła wraz z danymi kontaktowymi. Poniżej znajduje się obszar z przydzielonymi zadaniami dla użytkownika zawierający tytuł, opis, datę rozpoczęcia i zakończenia. Oprócz tego usytuowane są przyciski wyboru, mówiące o stanie aktualnego zadania - czy jest w trakcie robienia, czy został wykonany, sprawdzony i zatwierdzony. Dodatkowo zadania, które mają być wykonywane w bieżącym dniu, zostają zaznaczone zielonym tłem. Natomiast zadania niewykonane lub do poprawy określone są czerwonym tłem.



Nazwa	Opis	Dzień rozpoczęcia	Dzień zakończenia	Skończone	Sprawdzone	Potwierdzone
zadanie 4_1	opis 4_1	2015-12-08	2015-12-09	<input checked="" type="checkbox"/>		
		0000-00-00	0000-00-00	<input checked="" type="checkbox"/>		
		0000-00-00	0000-00-00	<input type="checkbox"/>		



Rysunek 22 Widok na zadania od strony pracownika



## 8.7 Obserwowanie projektu

Zakładka „Obserwowane” z panelu użytkownika przeznaczona jest dla osób, które zleciły stworzenie projektu i chciałyby mieć wgląd w aktualnie wykonywaną pracę. Jeśli klient nie widzi projektu w tej sekcji, powinien skontaktować się z managerem w celu dodania go do tej grupy.

Po wejściu w szczegóły projektu, klient dostaje informacje dotyczące projektu i osoby, która zajmuje się jego prowadzeniem – tak jak w przypadku panelu dla pracownika. Oprócz tego, dostaje listę wykonanych zadań przez pracownika i zatwierdzonych przez managera. Zadaniem klienta jest sprawdzenie wykonanych funkcjonalności, czy są zgodne z jego ideą. Jeśli tak to za pomocą przycisku potwierdza zgodność, a jeżeli nie, to przy pomocy dodatkowego okna zgłasza swoje uwagi. Informacja przekazywana jest bezpośrednio do programisty, który po poprawieniu zadania określa wykonanie zadania i wysyła je do sprawdzenia klientowi. Pozwala to na bieżącą reakcję w projekcie.

Zadania		
Nazwa	Opis	
zadanie 2bcd	opis 2222	 

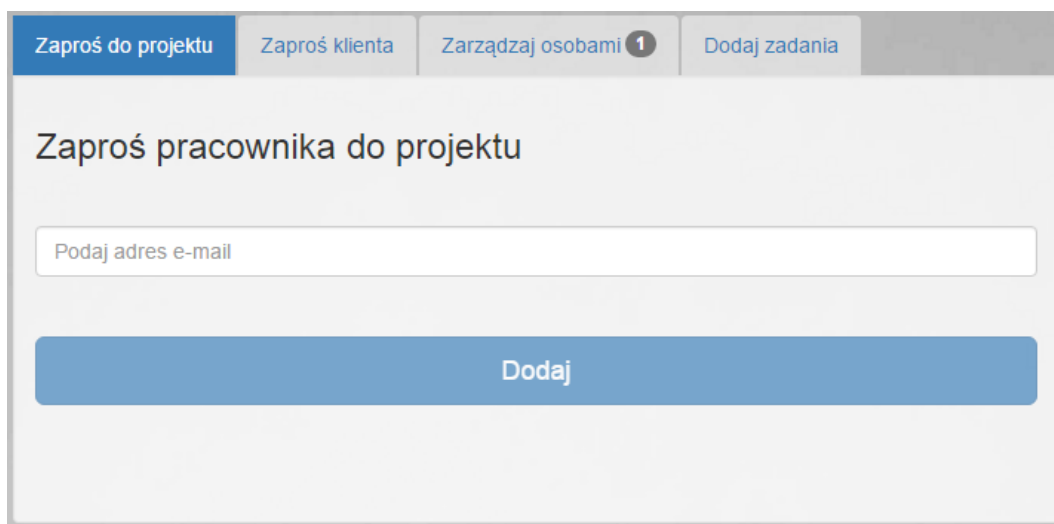
*Rysunek 23 Widok na zadania od strony klienta*

## 8.8 Zapraszanie pracowników i klientów do projektu

Osoba zarządzająca projektem dostaje przydatne funkcjonalności takie jak możliwość zaproszenia pracowników i klientów. Oprócz zapraszania posiada ewentualność zarządzania nimi poprzez wykluczenie w przypadku konfliktu lub innych nieprzyjemnych sytuacji.

Osoba ma do dyspozycji 3 zakładki, pozwalające kierować osobami w projekcie. Pierwsza z nich pozwala zapraszać osoby w charakterze pracowników. Odbywa się to przez wpisywanie adresu e-mail w pole a system automatycznie wyszukuje, czy taka osoba istnieje w bazie czy nie. Jeśli istnieje, to ramka formularza zostaje pokolorowana na zielono. Jeśli takiego adresu nie ma to ramka będzie koloru czerwonego. Od razu przycisk dodawania zostanie odblokowany i po jego naciśnięciu zostanie wysłane zaproszenie na wpisany adres e - mail. Jeśli osoba o takim adresie już istnieje w bazie, to zostanie wysłane powiadomienie

na adres e - mail o dodaniu do projektu. Jeśli jednak takiego pracownika nie ma, to system wygeneruje automatyczne dane logowania, które po zalogowaniu będzie można zmienić. Analogicznie sytuacja wygląda z dodawaniem klienta w drugiej zakładce.



Zaproś do projektu   Zaproś klienta   Zarządzaj osobami **1**   Dodaj zadania

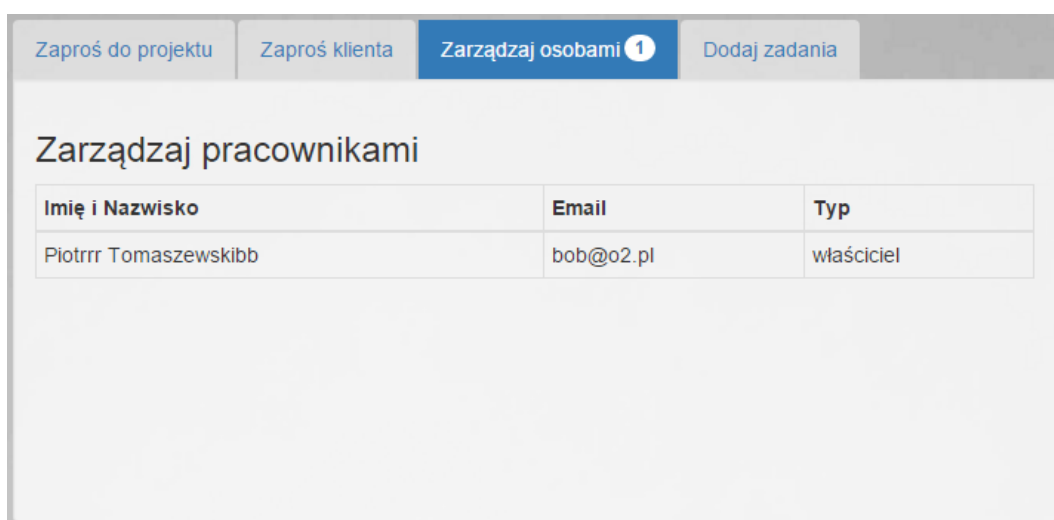
### Zaproś pracownika do projektu

Podaj adres e-mail

**Dodaj**

*Rysunek 24 Funkcjonalność pozwalająca zapraszać pracowników do projektu*

W zakładce służącej do zarządzania osobami, manager może zobaczyć informacje odnośnie dodanych użytkowników – ich imię i nazwisko, dane kontaktowe i typ. Wszyscy inni, poza właścicielem mogą być przez niego wykluczeni z projektu. Wystarczy kliknąć krzyżyk przy imieniu i nazwisku osoby.



Zaproś do projektu   Zaproś klienta   Zarządzaj osobami **1**   Dodaj zadania

### Zarządzaj pracownikami

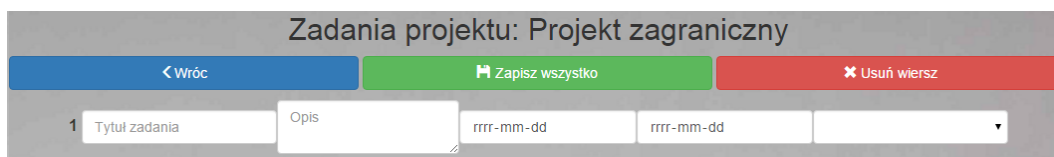
Imię i Nazwisko	Email	Typ
Piotrrr Tomaszewskibb	bob@o2.pl	właściciel

*Rysunek 25 Funkcjonalność pozwalająca na zarządzanie osobami przydzielonymi do projektu*

## 8.9 Tworzenie zadań w projekcie

Poza funkcjonalnością dotyczącą zarządzania użytkownikami, manager może, a nawet musi określać zadania do wykonania, według idei i przeznaczenia systemu. Każde zadanie musi być dobrze przedstawione wraz z określeniem celu, aby programista bez problemów mógł zrozumieć co tak naprawdę ma zrobić.

Do wyznaczania zadań służy zakładka „Dodaj zadania” otwierająca obszar roboczy. Zawsze na początku utworzone jest pierwsze zadanie, które ma charakter domyślny i którego nie można usunąć. W górnej części można zaobserwować trzy przyciski, z których pierwszy jest przeznaczony do powrotu do strony poprzedniej, drugi do zapisu dokonanych zmian, a trzeci do usunięcia zadania. Warto podkreślić, że zadanie utworzone, lub wyedytowane jest automatycznie zapisywane. Jednak warto dla bezpieczeństwa co jakiś czas zapisywać cały zestaw zadań, albo przed samym wyjściem z obszaru roboczego.



Panel górny obszaru do tworzenia zadań. Tytuł: Zadania projektu: Projekt zagraniczny. Przyciski: < Wróć (niebieski), Zapisz wszystko (zielony), ✕ Usuń wiersz (czerwony). Pola formularza: 1 Tytuł zadania, Opis, rrrr-mm-dd, rrrr-mm-dd, menu rozwijane.

**Rysunek 26 Panel górny obszaru do tworzenia zadań**

Aby dodać zadanie, wystarczy kliknąć w przycisk znajdujący się na dole strony. Jednak zanim zostanie stworzone zadanie, powinno się zaznaczyć istniejące zadanie, pod którym ma się utworzyć nowe. System pozwala nam również ustawić hierarchię zadań – jedno zadanie może mieć pod sobą kilka zadań, przy czym ważna jest zgodność dat, aby początek zadania zgadzał się z początkiem pierwszego podzadania, oraz koniec zadania z końcem ostatniego podzadania. Z zadania można zrobić podzadanie tylko raz. Istnieje możliwość powrotu. Nawigacja odbywa się za pomocą strzałek znajdujących się po prawej stronie zadania. Aby zrobić podzadanie podzadania, należy stworzyć zadanie, następnie przekształcić je w podzadanie, potem zaznaczyć podzadanie i utworzyć do niego zadanie i na końcu zmienić utworzone zadanie w podzadanie.

1	Tytuł zadania	Opis	rrrr-mm-dd	rrrr-mm-dd	▼
1.1	Tytuł zadania	Opis	rrrr-mm-dd	rrrr-mm-dd	▼
1.1.1	Tytuł zadania	Opis	rrrr-mm-dd	rrrr-mm-dd	▼

Dodaj kolejny wiersz

***Rysunek 27 Hierarchiczna prezentacja tworzenia zadań***

Nie wszystkie zadania można ustawiać hierarchicznie. Jeśli jakieś zadanie jest rodzicem dla podzadań, wtedy nie ma już możliwości wykonania przesunięcia w hierarchii. Zadania wykonane i zatwierdzone są automatycznie blokowane przez system. Jeśli programista zacznie wykonywać jakieś zadanie, jest ono również blokowane, aby uniknąć niespodziewanej zmiany treści. Jeśli jakieś zadanie nie jest wykonane, a minął czas na to przeznaczony, to oznaczone zostanie czerwonym tłem. Jeśli zadanie powinno być wykonywane w bieżącym terminie, to jego tło ma kolor zielony. Zadania skończone przez pracownika będą miały żółte tło wraz z przyciskiem zatwierdzenia zgodności wykonania zadania.

## **8.10 Kontakt z administratorem**

Aby skontaktować się z administratorem aplikacji lub baz danych użytkownik powinien kliknąć w kopertę, w menu znajdującym się w górnym obszarze aplikacji. Pojawi się okno, w którym należy podać temat problemu, adres email na który ma zostać wysłana odpowiedź oraz treść zgłoszenia. Po uzupełnieniu wszystkich pól wiadomość zostanie przesłana do administratora w celu podjęcia interwencji.

## 9. Podsumowanie

Budowa zintegrowanego systemu zarządzania projektem informatycznym, którego implementacja i specyfikacja była celem pracy dyplomowej, została wykonana. Na samym końcu przeprowadzono testy manualne świadczące o prawidłowym funkcjonowaniu aplikacji. Zastosowane technologie, języki programowania i algorytmy przyspieszyły w znaczny sposób tworzenie aplikacji. Była to również bardzo dobra szkoła pozwalająca na przećwiczenie najnowszych sposobów programowania, wykorzystywanych w nowoczesnych projektach. Ważne jest, aby poznawać i używać najnowszych rozwiązań w ciągle zmieniającej się branży IT.

W związku z tym, że system tworzony był przez jedną osobę, chciałem go wyróżnić pod względem funkcjonalnym jak i wizualnym od dostępnych, gotowych i profesjonalnych rozwiązań. Stworzona aplikacja jest podstawą do dalszego rozwoju pracy. Zastosowanie modelu MVC pozwala na prostsze i szybsze zastosowanie ulepszeń oraz uaktualnianie funkcjonalności z biegiem czasu. Osobistym sukcesem uważam szybkie zaznajomienie się z frameworkiem Codeigniter. Wcześniej pisane przeze mnie aplikacje w PHP były amatorskie. Nie zwracałem uwagi na wzorce i na obiektowość języka. Moim pierwszym zamierzeniem było stworzenie back-endu za pomocą Java Enterprise Edition z wykorzystaniem Hibernate, Vaadin, JPA, Oracle. Jednak to założenie okazało się zbyt ambitne, a presja czasu spowodowała, że musiałem skorzystać z języka PHP i słusznie bo efekt jest zadowalający.

Do zalet systemu mogę zaliczyć przede wszystkim szybkość działania. Nie posiada ona w sobie skomplikowanych operacji, przez co nie wymaga długiego czasu przetwarzania. Przede wszystkim aplikacja działa w pełni poprawnie. Wykonałem wszystkie zamierzone funkcjonalności. Podczas testowania manualnego system zachowuje się według zaplanowanych działań. W przypadku problemów wyświetlane są odpowiednie komunikaty.

Jednym z atutów jest nade wszystko możliwość rozwijania systemu przez osoby, które zainstalują to oprogramowanie na swoim serwerze. Dostęp do kodu daje możliwość pisania specjalnych pluginów lub wtyczek pod część bazową, stworzoną w ramach pracy dyplomowej.

Warto zwrócić uwagę na fakt, że dostęp do plików systemu jest jawny i niekontrolowany. Może on skutkować nielegalnym przywłaszczeniem modułów i rozwiązań w celach komercyjnych. Jest to jedna z dostępnych wad oprogramowania opartego na tak zwanym wolnym oprogramowaniu.

## 9.1 Testy manualne

Zdarzenie	Rezultat
Użytkownik poda zły login lub hasło.	Pojawił się komunikat mówiący o nieprawidłowym logowaniu z możliwością przypomnienia hasła.
Użytkownik nie uzupełnił wszystkich wymaganych pól przy rejestracji.	Pojawił się komunikat o konieczności uzupełnienia pól, przy których znajduje się symbol gwiazdki (*).
Użytkownik nie prawidłowo wpisał dane przy zmianie hasła.	System powiadomił użytkownika o nieprawidłowych danych niezbędnych do zmiany hasła do konta zalogowanego użytkownika.
Użytkownik pozostawił wymagane puste pole przy edycji	Pojawił się komunikat o konieczności uzupełnieniu pól, przy których znajduje się symbol gwiazdki (*).
Adres zapraszanego pracownika nie istnieje w bazie.	System poinformował użytkownika o braku podanego adresu e-mail w bazie. Zaproponował możliwość wysłania zaproszenia do projektu na podany adres e-mail.
Programista nie wykonał zadania w terminie.	Zadanie zostało zaznaczone przez system czerwonym tłem.
Manager chce usunąć zadanie bez wcześniejszego zaznaczenia	Pojawił się komunikat informujący o konieczności zaznaczenia wybranego zadania, by potem usunąć go i wszystkie jego dzieci, o ile istnieją.
Usunięcie pracownika z projektu	System bez ostrzeżenia usuwa osobę z projektu. Istnieje możliwość ponownego wprowadzenia przez wysłanie zaproszenia na adres e-mail

*Tabela 3 Tabela z możliwymi zdarzeniami wraz z rezultatami reakcji systemu*

## 9.2 Możliwości rozwoju

System jest na tyle uniwersalny, że jego kierunki rozwoju nie muszą iść wcale w branżę informatyczną. Ważne jest, żeby zachować pierwotny stan aplikacji, z wszystkimi podstawowymi funkcjami stworzonymi przeze mnie. Na pewno najważniejszą możliwością rozwoju byłoby poprawienie komunikacji pomiędzy pracownikami a managerem. Aktualnie system nie obsługuje wewnętrznego systemu wiadomości. Jedyna możliwa komunikacja pomiędzy pracownikami to droga mailowa albo za pośrednictwem prywatnego firmowego komunikatora.

Kolejną funkcją możliwą do wykonania jest przeznaczenie skończonych projektów do archiwizacji. Z czasem, gdy projektów będzie przybywać, lista projektów będzie coraz dłuższa, czego wynikiem będzie trudniejsze odnajdywanie starych niedokończonych projektów pośród już zakończonych. Aktualne rozwiązanie wyświetla zadania od najmłodszych do najstarszych, biorąc pod uwagę datę utworzenia.

Pomocną opcją dla tworzących projekt jest wprowadzenie opisu projektu, by móc zaznajomić się z jego ideą i tematyką. Opisu mógłby dokonywać manager, który tworzy projekt. Również przestrzeń dotycząca nazwy projektu mogłaby być edytowalna, na wypadek popełnienia literówki, w trakcie zakładania projektów.

Ciekawym rozwiązaniem jest również sprawdzanie pracowników, ile w danym miesiącu przepracowano godzin nad konkretnym projektem i ile wykonali średnio zadań. Będzie to dobra inicjatywa, pozwalająca zaangażowanym osobom wynagrodzić ich trud. Pozwoli to na zmotywowanie osób i wprowadzenie zdrowej rywalizacji.

Ostatnim i chyba najtrafniejszym elementem rozwoju systemu jest interakcja z systemami zarządzania treścią, typu GitHub albo Bitbucket. Wtedy osoba zarządzająca nie tylko ma wgląd w wykonane zadanie, ale może zobaczyć kod, który napisał programista, w celu jego analizy i zapewnienia wysokiej jakości kodu.

### **9.3 Zakończenie**

Zarządzanie projektami to nowa gałąź w dziedzinie informatyki. Powstaje coraz więcej szkoleń, przedmiotów na uczelniach, w celu przygotowania młodych pracowników do pracy w taki sposób. Coraz częściej duże firmy same kierują pracowników z wieloletnim stażem pracy by przekwalifikowały się na coraz bardziej popularny sposób pracy. Techniki takie jak SCRUM, Agile, PRINCE2 to tylko nieliczne z najpopularniejszych podejść zarządzania. Rozwinięcie zaproponowanego w pracy systemu o te techniki może być jednym ze sposobów jego rozwoju. Bardzo często zdarza się tak, że projekt zmienia się wraz z wizją klienta. Wtedy trzeba dostosować się do nowych potrzeb, ponieważ taka jest wizja osoby, która za bazowy produkt i jego rozwój płaci duże pieniądze. Jednak czy każdy jest w stanie dostosować się do takiego korporacyjnego trybu pracy? To zależy od osoby i jej indywidualnych predyspozycji. Myślę, że każdy czytający tę pracę powinien sobie na to pytanie sam odpowiedzieć w zgodzie z samym sobą i swoim umysłem.

## Bibliografia

1. Bibeault B., Resig J., *Tajemnice JavaScriptu. Podręcznik ninja*, Wydawnictwo HELION, Gliwice 2014.
2. Bruegge B., Dutoit A. H., *Inżynieria oprogramowania w ujęciu obiektowym*, Wydawnictwo HELION, Gliwice 2011.
3. Duckett J., *HTML i CSS. Zaprojektuj i zbuduj witrynę WWW*, Wydawnictwo HELION, Gliwice 2014.
4. Duckett J., *JavaScript i jQuery. Interaktywne strony WWW dla każdego*, Wydawnictwo HELION, Gliwice 2015.
5. Gamma E., Helm R., Johnson R., Vlissides J. M., *Wzorce projektowe. Elementy oprogramowania obiektowego wielokrotnego użytku*, Wydawnictwo HELION, Gliwice 2010.
6. Lis M., *Tworzenie bezpiecznych aplikacji internetowych*, Wydawnictwo HELION, Gliwice 2014.
7. Martin C. R., *Czysty kod. Podręcznik dobrego programisty*, Wydawnictwo HELION, Gliwice 2010.
8. Sonmez J., *Sprawny programista*, Wydawnictwo HELION, Gliwice 2015.
9. Wrotek W., *CSS3. Zaawansowane projekty*, Wydawnictwo HELION, Gliwice 2015.



## Netografia

1. *Co to jest Microsoft Project i do czego służy?* <http://project.komako.pl/msproject.html> (Data odczytu: 15 grudnia 2015).
2. *Dokumentacja – Bootstrap* <http://getbootstrap.com/> (Data odczytu: 7 grudnia 2015).
3. *Diagram sekwencji*, <http://www.michalwolski.pl/diagramy-uml/diagram-sekwencji/> (Data odczytu: 3 stycznia 2016).
4. *Jira – na co to, po co, komu to potrzebne i jak to coś skonfigurować na serwerze VPS*, <https://piktmaaitw.wordpress.com/2014/05/26/jira-na-co-to-po-co-komu-to-potrzebne-i-jak-to-cos-skonfigurowac-na-serwerze-vps/> (Data odczytu: 17 grudnia 2015).
5. *jQuery*, <http://kursjs.pl/kurs/jquery/jquery.html> (Data odczytu: 6 grudnia 2015).
6. *PHP dokumentacja użytkownika*, <http://php.net/manual/en/> (Data odczytu: 5 grudnia 2015).
7. *Podręcznik użytkownika CodeIgniter wersja 2.2.1* <http://podrecznik.codeigniter.org/pl/index.html> (Data odczytu: 17 września 2015).
8. *Projektowanie Systemów Komputerowych - UML, OCL, Wzorce Projektowe*, <http://brasil.cel.agh.edu.pl/~09sbfraczek/> (Data odczytu: 2 października 2015).
9. *Trello i zarządzanie zadaniami*, <http://productvision.pl/2014/trello-i-zarzadzanie-zadaniami> (Data odczytu: 15 grudnia 2015).
10. *Wzorzec MVC w PHP*, <http://ferrante.pl/frontend/php/wzorzec-mvc-w-php/> (Data odczytu: 5 grudnia 2015).

## Spis ilustracji

Rysunek 1 Schemat działania wzorca MVC .....	15
Rysunek 2 Diagram przypadków użycia .....	24
Rysunek 3 Diagram związków encji .....	25
Rysunek 4 Diagram DFD kontekstowy .....	27
Rysunek 5 Diagram DFD systemów .....	28
Rysunek 6 Diagram ELH dla obiektu Users .....	29
Rysunek 7 Diagram ELH dla obiektu Project .....	30
Rysunek 8 Diagram ELH dla obiektu Type .....	30
Rysunek 9 Diagram ELH dla obiektu Tasks_proj .....	31
Rysunek 10 Panel powitalny narzędzia phpMyAdmin .....	39
Rysunek 11 Narzędzie PhpStorm do zarządzania kodem źródłowym systemu .....	40
Rysunek 12 Pakiet zawierający pliki aplikacji oraz plik instalacyjny bazy danych .....	40
Rysunek 13 Tworzenie nowej bazy danych pod tabele systemu .....	41
Rysunek 14 Importowanie pliku konfiguracyjnego bazy danych .....	41
Rysunek 15 Struktura tabel po zaimportowaniu pliku SQL .....	42
Rysunek 16 Po prawej: pakiet instalacyjny na komputerze; po lewej: pakiet instalacyjny na serwerze .....	42
Rysunek 17 Panel logowania do systemu zarządzania projektami .....	44
Rysunek 18 Panel rejestracyjny dla użytkowników .....	45
Rysunek 19 Panel zarządzający utworzonymi projektami .....	46
Rysunek 20 Panel zmiany danych i hasła .....	47
Rysunek 21 Okno z formularzem do tworzenia nowego projektu .....	48
Rysunek 22 Widok na zadania od strony pracownika .....	48
Rysunek 23 Widok na zadania od strony klienta .....	49
Rysunek 24 Funkcjonalność pozwalająca zapraszać pracowników do projektu .....	50
Rysunek 25 Funkcjonalność pozwalająca na zarządzanie osobami przydzielonymi do projektu .....	50
Rysunek 26 Panel górny obszaru do tworzenia zadań .....	51
Rysunek 27 Hierarchiczna prezentacja tworzenia zadań .....	52

## Spis tabel

Tabela 1 Określenie zbiorów encji .....	35
Tabela 2 Tabela atrybutów encji bazy danych .....	36
Tabela 3 Tabela z możliwymi zdarzeniami wraz z rezultatami reakcji systemu .....	54