Dokumentacja implementacyjna

„Aplikacja wspierająca inwentaryzację i zarządzanie pracownikami w Pubie Drewutnia”

Politechnika Krakowska im. Tadeusza Kościuszki, Wydział Inżynierii Elektrycznej i Komputerowej w Krakowie.

Jakub Pranica
Robert Trojan
Grupa 33i

1. Wprowadzenie 3
2. Przegląd zawartości Dokumentacji implementacyjnej 3
3. Słownik pojęć 4
4. Interfejsy 4
5. Wymagania funkcjonalne 6
6. Wymagania niefunkcjonalne 14
7. Technologie 15
8. Diagram klas 18
9. Słownik klas 19
10. Architektura systemu 20
11. Baza danych 20
12. Schemat bazy danych 22
13. Źródła 23

1. Wprowadzenie

1.1 Niniejsza dokumentacja stanowi opis kwestii technicznych aplikacji. Jej celem jest umożliwienie zapoznania się z technologią oraz ułatwienie rozbudowy aplikacji.

2. Przegląd zawartości Dokumentacji implementacyjnej

2.1 Poniżej wyszczególnione zostaną wszystkie rozdziały wraz z opisem ich zawartości.

2.2 Słownik pojęć (*rozdział 3*) - w tym rozdziale zostaną opisane i wyjaśnione pojęcia, które mogą być używane potocznie, są pojęciami technicznymi danej dziedziny lub po prostu ich niezrozumienie może utrudnić zrozumienie dokumentu.

2.9 Interfejsy (*rozdział 4*) - w tym rozdziale opisane zostaną interfejsy tworzonego systemu. Zakłada się ich dopasowanie do indywidualnych preferencji użytkowników.

2.10 Wymagania funkcjonalne (*rozdział 5*) - w tym rozdziale wyspecyfikowane zostaną wymagania funkcjonalne tworzonego systemu.

2.11 Wymagania niefunkcjonalne (*rozdział 6*) - w tym rozdziale wyspecyfikowane zostaną wymagania niefunkcjonalne tworzonego systemu.

2.12 Technologie (*rozdział 7*) - w tym rozdziale opisane zostaną technologie, które zostaną użyte do stworzenia aplikacji..

2.13 Diagram klas *(rozdział 8)* - statyczny diagram strukturalny, przedstawiający strukturę systemu w modelach obiektowych przez ilustrację struktury klas i zależności między nimi.

2.14 Słownik klas *(rozdział 9)* - opis każdej z klas.

2.15 Architektura systemu *(rozdział 10)* - opis zastosowanej architektury klient-serwer-baza.

2.16 Baza danych *(rozdział 11)* - wyjaśnienie wyboru technologii oraz opis zabiegów optymalizacyjnych.

2.15 Źródła (*rozdział 16*) - w tym rozdziale zawarte zostaną informację o źródłach wykorzystanych przy tworzeniu tego dokumentu.

3. Słownik pojęć

Uwaga! Poniższy słownik pojęć został ograniczony do słownictwa występującego w tej dokumentacji - pełne repozytorium dostępne jest w dokumentacji projektowej.

3.1 GUI - (od angielskiego *graphical user interface*) - inaczej interfejs graficzny. Określa sposób prezentacji informacji przez komputer oraz interakcji z użytkownikiem, polegający na rysowaniu i obsługiwaniu widżetów.

3.2 Stan kasy - ilość pieniędzy znajdujących się w danej chwili w kasie. W Pubie Drewutnia liczony zaraz po otwarciu oraz zaraz po zamknięciu lokalu.

3.3 Input - interaktywny element graficznego interfejsu pozwalający na wprowadzenie danych z klawiatury.

3.4 Raport miesięczny - raport przygotowywany przez menadżera pod koniec każdego miesiąca. Przeznaczony jest dla właściciela pubu. Zawiera informacje statystyki i wykresy dotyczące finansów z całego miesiąca oraz dodatkowe uwagi i sugestie.

3.5 Alert - Inaczej powiadomienie. W przypadku opisywanego systemu - okno aplikacji wyświetlające ważną wiadomość.

Słownik klas znajduje się w dalszej części dokumentu.

4. Interfejsy

4.1 Poniższa część dokumentu opisuje interfejsy tworzonego systemu.

4.2 Interfejsy użytkowników - zostaną zrealizowany w formie graficznego interfejsu (*słownik pojęć 3.1*). Dostępny on będzie wyłącznie z poziomu aplikacji klienckiej. Obsługa GUI (*słownik pojęć 3.1*) będzie możliwa za pomocą myszki i klawiatury. Najważniejszymi cechami tworzonego interfejsu będą przejrzystość, minimalizm i prostota. Każdy użytkownik po uruchomieniu aplikacji zobaczy stronę startową zawierającą opcje logowania do konta. Będzie ona zawierała pola do podania nazwy użytkownika oraz hasła. Po poprawnym zalogowaniu interfejs będzie różny w zależności od tego kto się zalogował. Menadżer zobaczy inne okno aplikacji niż pracownik.

4.3 Interfejs pracownika po zalogowaniu - pierwszym oknem po zalogowaniu jakie zobaczy pracownik będzie okno, w którym aplikacja wymusi wprowadzenie stanu kasy (*słownik pojęć 3.2*). Wyświetlą się liczby symbolizujące nominały monet oraz banknotów. Obok każdego z nominałów dostępny będzie element typu input (*słownik pojęć 3.3*), do którego będzie można wprowadzić ilość banknotów

lub monet o nominale wyświetlonym obok input'a (*słownik pojęć 3.3*). Po wpisaniu wszystkich wartości pracownik będzie mógł zatwierdzić wpisane dane przyciskiem „zatwierdź". Po zatwierdzeniu zostanie wyświetlone to samo okno aplikacji jeszcze raz, jednak znikną wszystkie wartości wpisane do elementów typu input (*słownik pojęć 3.3*). Zadaniem pracownika będzie ponowne policzenie ilości każdego rodzaju banknotu i każdego rodzaju monety oraz ponowne wpisanie odpowiednich liczb do aplikacji. Aplikacja porówna dane wpisane za pierwszym i drugim razem. Jeżeli nastąpiłby błąd aplikacja poinformuje o nim pracownika, wyświetli przy jakim nominale się pomylił i poprosi, aby policzył jeszcze raz i wpisał ostateczną ilość. Po poprawnym przejściu procedury aplikacja wyświetli główne okno. W głównym oknie pracownik będzie miał możliwość obserwowania jak nabite na kasę fiskalną produkty wyświetlane są w aplikacji jako sprzedane. W oknie głównym pracownik będzie miał również możliwość zostawienia notatki dla następnego pracownika lub dla menadżera, będzie mógł też przejść do okna inwentaryzacji oraz wylogować się. Okno inwentaryzacji pozwoli na wprowadzanie danych o ilości konkretnych produktów. Wykorzystane w tym celu będą pola tekstowe wyświetlające nazwę produktu oraz inputy (*słownik pojęć 3.3*) pozwalające wprowadzić ilość magazynową danego produktu.

4.4 Interfejs menadżera po zalogowaniu - pierwszym oknem po zalogowaniu jakie zobaczy menadżer będzie od razu okno główne aplikacji. W oknie tym menadżer będzie miał wgląd do wiadomości zostawionych przez pracowników (notatek). Będzie miał również możliwość sprawdzenia sprzedanych w danym dniu produktów. Kolejnymi możliwościami będzie sprawdzenie ilości przepracowanych przez pracowników godzin, sprawdzenie stanu magazynu oraz kasy. Menadżer będzie z tego okna mógł również przejść do okna sporządzenia raportu miesięcznego (*słownik pojęć 3.4*). Oczywiście zostanie dodana również możliwość wylogowania się z konta.

4.5 Interfejsy sprzętowe - wymogiem do korzystania z tworzonego systemu będzie komputer personalny (PC) lub laptop z systemem operacyjnym Windows lub OSX. Jedynymi wymogami urządzeń, na których będzie uruchamiana aplikacja jest obsługa połączeń sieciowych oraz aplikacji napisanych w języku JAVA 9 lub nowszym.

4.6 Interfejsy programowe - komunikacja pomiędzy bazą danych a aplikacją serwerową składać się będzie zarówno z odczytu jak i modyfikacji danych. Wymagane jest zatem, aby aplikacja serwerowa miała możliwość łączenia się z bazą przechowującą dane potrzebne do funkcjonowania aplikacji.

4.7 Interfejsy komunikacyjne - komunikacja pomiędzy aplikacją kliencką a serwerem będzie odbywała się przy pomocy protokołu HTTP. W przypadku komunikacji pomiędzy bazą danych a serwerem, która polega na odczytywaniu i

modyfikowaniu zawartości bazy danych, nie określa się konkretnego sposobu komunikacji, tak długo jak będzie on poprawnie wykonywał swoje zadanie.

5. Wymagania funkcjonalne

5.1 Poniższa część dokumentu zawiera specyfikację wymagań funkcjonalnych tworzonego systemu. Diagram wymagań funkcjonalnych został umieszczony w dokumentacji projektowej.

5.2.1 Nazwa - Logowanie do systemu

5.2.2 ID - LDS

5.2.3 Cel - Umożliwienie dowolnemu użytkownikowi posiadającemu konto zalogowanie się do niego, aby móc korzystać z możliwości systemu.

5.2.4 Opis działania - Użytkownik musi mieć możliwość opcji zalogowania do swojego konta. W tym celu powinien mieć obowiązek podania swojej nazwy użytkownika oraz swojego hasła do konta. Użytkownik po wpisaniu danych powinien je zatwierdzić przyciskiem. System powinien poinformować użytkownika o rezultacie próby zalogowania poprzez wyświetlenie odpowiedniego komunikatu tekstowego.

5.2.5 Użytkownicy, mogący wykorzystać - menadżer, pracownik

5.2.6 Powiązane wymagania funkcjonalne - Rejestracja kont pracowników

5.3.1 Nazwa - Rejestracja kont pracowników

5.3.2 ID - RKP

5.3.3 Cel - Umożliwienie utworzenia nowego konta.

5.3.4 Opis działania - Podczas zatrudnienia nowych pracowników powinni oni mieć dostęp do swojego indywidualnego konta, które musi zostać w tym celu stworzone. Tworzyć nowe konto będzie mógł tylko menadżer za pośrednictwem swojego konta mającego odpowiednie uprawnienia. Podczas tworzenia konta, menadżer będzie mógł również określić uprawnienia tworzonego konta (uprawnienia menadżera lub pracownika)

5.3.5 Użytkownicy mogący wykorzystać - menadżer

5.3.6 Powiązane wymagania funkcjonalne - Logowanie do systemu

5.4.1 Nazwa - Zmiana hasła

5.4.2 ID - ZH

5.4.3 Cel - Umożliwienie zmiany hasła. Może wystąpić potrzeba zmiany hasła, gdy wystąpi podejrzenie, że obecne hasło dostało się w niepowołane ręce.

5.4.4 Opis działania - W ustawieniach konta będzie możliwość wybrania opcji zmiany hasła. Aby zmienić hasło trzeba będzie podać obecne hasło, a następnie wpisać nowe. Nowe hasło trzeba będzie potwierdzić wpisując je jeszcze raz.

5.4.5 Użytkownicy mogący wykorzystać - menadżer, pracownik

5.4.6 Powiązane wymagania funkcjonalne - Logowanie do systemu

5.5.1 Nazwa - Przypomnienie hasła

5.5.2 ID - PH

5.5.3 Cel - W razie, gdy użytkownik aplikacji zapomni swojego hasła będzie istniała możliwość przypomnienia go.

5.5.4 Opis działania - Podczas wpisywania hasła będzie możliwość kliknięcia opcji „przypomnij mi hasło”. Po wybraniu tej opcji użytkownik będzie musiał wpisać swoją nazwę użytkownika, gdy to zrobi na adres email powiązany z nazwą użytkownika zostanie wysłany mail zawierający hasło do logowania.

5.5.5 Użytkownicy mogący wykorzystać - menadżer, pracownik

5.5.6 Powiązane wymagania funkcjonalne - Logowanie do systemu

5.6.1 Nazwa - Zmiana uprawnień zarejestrowanych kont

5.6.2 ID - ZUZK

5.6.3 Cel - Możliwość nadania zarejestrowanemu wcześniej kontu więcej praw lub odebranie tych praw.

5.6.4 Opis działania - Zmieniać uprawnienia zarejestrowanego konta może tylko użytkownik aplikacji z uprawnieniami menadżera. Dostępne możliwości uprawnień to „pracownik” oraz „menadżer”.

5.6.5 Użytkownicy mogący wykorzystać - menadżer

5.6.6 Powiązane wymagania funkcjonalne - Rejestracja kont pracowników, Logowanie do systemu,

5.7.1 Nazwa - Obliczanie przepracowanych godzin

5.7.2 ID - OPG

5.7.3 Cel - Funkcja systemu ma za zadanie zliczać czas jaki przepracował pracownik, w każdym dniu miesiąca. Umożliwi to obliczenie wysokość wynagrodzenia dla każdego z pracowników.

5.7.4 Opis działania - Obliczanie przepracowanych godzin nie będzie wymagało, żadnej ingerencji ani ze strony pracowników, ani menadżera. System automatycznie (w tle) będzie liczył czas pracy każdego z pracowników, a czas ten liczony będzie od chwili zalogowania do systemu przez danego pracownika aż do chwili wylogowania.

5.7.5 Użytkownicy mogący wykorzystać - brak (działa automatycznie w tle)

5.7.6 Powiązane wymagania funkcjonalne - Logowanie do systemu, Wyświetlanie przepracowanych godzin

5.8.1 Nazwa - Wyświetlanie przepracowanych godzin

5.8.2 ID - WPG

5.8.3 Cel - Ta funkcja systemu jest przeznaczona do wyświetlenia w specjalnym panelu informacji o przepracowanych godzinach osobno dla każdego z pracowników.

5.8.4 Opis działania - W specjalnym panelu będzie możliwość określenia z jakiego okresu czasu ma zostać pobrana, a następnie zsumowana ilość przepracowanych godzin. Dane wyświetlane będą w formie tabel. Będzie również możliwość stworzenia wykresu na podstawie wyświetlanej tabeli.

5.8.5 Użytkownicy mogący wykorzystać - menadżer

5.8.6 Powiązane wymagania funkcjonalne - Logowanie do systemu, Obliczanie przepracowanych godzin

5.9.1 Nazwa - Wprowadzanie informacji o stanie kasy

5.9.2 ID - WIoSK

5.9.3 Cel - Funkcja ta umożliwi pracownikom wprowadzanie informacji o stanie kasy w momencie przystąpienia przez nich do pracy oraz na koniec po zakończeniu pracy. Informacje te są potrzebne do kontrolowania ilości pieniędzy w kasie.

5.9.4 Opis działania - Po zalogowaniu się pracownika do systemu aplikacja wyświetli okno wprowadzania informacji o stanie kasy. Pracownik nie będzie mógł pominąć tej opcji. Jego zadaniem będzie przeliczyć ilość każdego rodzaju banknotu oraz każdego monety w kasie. W oknie wprowadzania informacji będzie

musiał wpisać policzone wartości po czym zatwierdzić całość przyciskiem znajdującym się w dolnej części okna aplikacji. Po zatwierdzeniu aplikacja poprosi o wpisanie ilości wszystkich banknotów oraz monet jeszcze raz, w celu uniknięcia pomyłki. Pracownik w tym momencie będzie miał obowiązek policzyć ilość pieniędzy jeszcze raz, a następnie ponownie wprowadzić ją do aplikacji i zatwierdzić przyciskiem. Jeżeli podane wartości będą poprawne, aplikacja przejdzie do kolejnego okna. Jeżeli jednak, w którymś z wypełnionych pól będzie błąd, wtedy pracownik zostanie poproszony o dokładne przeliczenie danej wartości banknotu lub monety (tego/tej, przy której pojawił się błąd) po raz trzeci.

5.9.5 Użytkownicy mogący wykorzystać - pracownik

5.9.6 Powiązane wymagania funkcjonalne - Powiadomienia dla menadżera o niezgodnościach

5.10.1 Nazwa - Pobieranie i przechowywanie informacji o sprzedanych produktach

5.10.2 ID - PiPIoSP

5.10.3 Cel - Aplikacja ma pobierać i przechowywać informacje o sprzedawanych produktach, aby móc kontrolować czy rzeczywisty stan kasy pokrywa się z obliczonym na podstawie sprzedaży.

5.10.4 Opis działania - Funkcja działa w tle zawsze, gdy użytkownik (pracownik lub menadżer) jest zalogowany. Informacje o sprzedaży są pobierane automatycznie dzięki integracji z kasą fiskalną. Wszystkie informacje o produktach nabitych na kasę fiskalną trafią do bazy danych aplikacji. Możliwa będzie również ręczna ingerencja w dane o sprzedanych produktach, aby móc skorygować ewentualne błędy jednakże, gdy pracownik wykorzysta taką możliwość, każdorazowo informacja o tym trafi do menadżera.

5.10.5 Użytkownicy mogący wykorzystać - brak (działa automatycznie w tle)

5.10.6 Powiązane wymagania funkcjonalne - Powiadomienia dla menadżera o niezgodnościach

5.11.1 Nazwa - Wprowadzanie informacji o stanie magazynu

5.11.2 ID - WIoSM

5.11.3 Cel - Ta funkcja systemu będzie umożliwiała przeprowadzenie inwentaryzacji.

5.11.4 Opis działania - Użytkownik chcąc przeprowadzić inwentaryzację będzie wybierał z menu aplikacji opisywaną funkcjonalność. Wyświetlone zostanie

wtedy okno inwentaryzacji, w którym aplikacja po kolej będzie prosiła użytkownika o przeliczenie ilości konkretnego produktu. Po wpisaniu ilości produktu, aplikacja będzie prosiła o ponowne przeliczenie i podanie ilości produktu jeszcze raz w celu uniknięcia pomyłki.

5.11.5 Użytkownicy mogący wykorzystać - pracownik

5.11.6 Powiązane wymagania funkcjonalne - Powiadomienia dla menadżera o niezgodnościach

5.12.1 Nazwa - Powiadomienia dla menadżera o niezgodnościach

5.12.2 ID - PDMoN

5.12.3 Cel - Celem tej funkcjonalności jest poinformowanie menadżera, o wszystkich niezgodnościach polegających na błędnym policzeniu towarów lub pieniędzy.

5.12.4 Opis działania - Aplikacja wyświetla alert (*słownik pojęć 3.5*) za każdym razem, gdy przy inwentaryzacji lub obliczaniu pieniędzy w kasie wchodzi inna niż obliczona przez aplikację ilość towarów lub pieniędzy. Alert widoczny jest tylko dla użytkownika z uprawnieniami menadżera. Pracownik wpisujący błędną ilość nie dowie się, że jest ona sprzeczna z wyliczeniami aplikacji.

5.12.5 Użytkownicy mogący wykorzystać - menadżer

5.12.6 Powiązane wymagania funkcjonalne - Wprowadzanie informacji o stanie kasy, Pobieranie i przechowywanie informacji o sprzedanych produktach, Wprowadzanie informacji o stanie magazynu

5.13.1 Nazwa - Generowanie raportów miesięcznych

5.13.2 ID - GRM

5.13.3 Cel - Funkcja ta ma za zadanie umożliwić wygenerowanie w dowolnym czasie raportów zawierających informacje z ostatniego miesiąca dotyczące stanu magazynu, sprzedaży, ilości przepracowanych godzin przez poszczególnych pracowników oraz informacji o finansach.

5.13.4 Opis działania - Menadżer będzie miał możliwość wybrania z menu aplikacji opcji generowania raportu. Po jej wyborze wyświetlone zostanie okno, w którym możliwe będzie skonfigurowanie raportu. Menadżer będzie mógł zdecydować co ma pojawić się w raporcie. Będzie miał również możliwość wybrania czy wygenerowany ma zostać raport w formie tylko tabel czy też dla niektórych tabel powinny zostać stworzone wykresy. Aplikacja da możliwość wybrania formy wykresów (np. kołowe, słupkowe itp.).

5.13.5 Użytkownicy mogący wykorzystać - menadżer

5.13.6 Powiązane wymagania funkcjonalne - Obliczanie przepracowanych godzin, Wprowadzanie informacji o stanie kasy, Pobieranie i przechowywanie informacji o sprzedanych produktach, Wprowadzanie informacji o stanie magazynu

5.14.1 Nazwa - Przechowywanie i wyświetlanie dokumentów

5.14.2 ID - PiWD

5.14.3 Cel - Funkcja ta ma za zadanie przechowywać skany (w tym zdjęcia) dokumentów, takich jak faktury, paragony, umowy oraz wszystkich innych plików z informacjami. Opisywana funkcjonalność powinna również pomagać w wygodnym wyszukiwaniu i wyświetlaniu przechowywanych dokumentów.

5.14.4 Opis działania - Menadżer będzie miał możliwość wybrania z menu aplikacji opcji o nazwie „Dokumenty”. Po otwarciu tej zakładki będzie miał możliwość wyszukania i przeglądania przechowywanych dokumentów. Będzie możliwość grupowania przechowywanych plików w folderach oraz dodawania im etykiet, dzięki którym będzie można łatwiej je odszukać. Oczywiście w zakładce „Dokumenty” będzie również możliwość wgrywania do aplikacji nowych plików. Mogą to być pliki z rozszerzeniami graficznymi lub tekstowymi. Aplikacja będzie obsługiwała najpopularniejsze rozszerzenia.

5.14.5 Użytkownicy mogący wykorzystać - menadżer

5.14.6 Powiązane wymagania funkcjonalne - brak

5.15.1 Nazwa - Tworzenie kopii zapasowej danych aplikacji

5.15.2 ID - TKZDA

5.15.3 Cel - Aplikacja da możliwość tworzenia kopia zapasowej wszystkich przechowywanych informacji w celu umożliwienia ich odzyskania w przypadku nieprzewidzianej awarii wiążącej się z utratą danych z dysku.

5.15.4 Opis działania - W ustawieniach aplikacji będzie możliwe zdefiniowanie jak często aplikacja ma automatycznie wykonać kopię wszystkich swoich danych. Będzie można wybrać opcję tworzenia automatycznej kopii zapasowej co określoną liczbę miesięcy, tygodni, dni lub godzin. Możliwe będzie również wybranie opcji tworzenia kopii po każdym wylogowaniu. Menadżer będzie mógł również zdefiniować jak długo przechowywana będzie każda z kopii.

5.15.5 Użytkownicy mogący wykorzystać - menadżer

5.15.6 Powiązane wymagania funkcjonalne - Przywracanie kopii zapasowej danych aplikacji

5.16.1 Nazwa - Przywracanie kopii zapasowej danych aplikacji

5.16.2 ID - PKZDA

5.16.3 Cel - Opcja umożliwi wczytanie wszystkich wcześniej wyeksportowanych danych z powrotem do aplikacji. Dzięki temu zawsze będzie możliwość przywrócenia ważnych danych.

5.16.4 Opis działania - W przypadku utraty danych lub w przypadku, gdy nastąpi chęć wczytania wcześniejszego stanu aplikacji będzie możliwe wczytanie stworzonej wcześniej kopii zapasowej. Kopia zapasowa będzie musiała być określonego formatu dlatego najlepiej, żeby była to nie edytowana kopia stworzona przez aplikację.

5.16.5 Użytkownicy mogący wykorzystać - menadżer

5.16.6 Powiązane wymagania funkcjonalne - Tworzenie kopii zapasowej danych aplikacji

5.17.1 Nazwa - Wprowadzenie do systemu - samouczek

5.17.2 ID - WDS

5.17.3 Cel - Będzie to funkcja pozwalająca nowym użytkownikom systemu na łatwe zapoznanie się z interfejsem graficznym oraz funkcjonalnościami systemu.

5.17.4 Opis działania - Przy pierwszym uruchomieniu aplikacji przez nowego użytkownika (menadżera lub pracownika) zostanie wyświetlone okno z polem tekstowym, w którym zostanie wyświetlona wiadomość powitalna samouczka. Poniżej będzie wyświetlony przycisk przejścia dalej. Po każdym jego kliknięciu w polu tekstowym pojawiać się będzie opis kolejnych funkcji systemu wraz ze strzałką pokazującą gdzie można je znaleźć. Po przejściu całego samouczka jego okno zniknie, a użytkownik będzie mógł przejść do obsługi aplikacji. Samouczek można będzie również włączać z menu aplikacji za każdym razem kiedy będzie potrzebny.

5.17.5 Użytkownicy mogący korzystać - menadżer, pracownik

5.17.6 Powiązane wymagania funkcjonalne - Logowanie do systemu

5.18.1 Nazwa - Zostawianie notatek

5.18.2 ID - ZN

5.18.3 Cel - Funkcja ta pozwoli pracownikowi lub menadżerowi napisania notatki, czyli krótkiej wiadomości. Notatka taka pozwoli przekazać najważniejsze informacje.

5.18.4 Opis działania - Użytkownik, który zechce zostawić notatkę będzie musiał przejść do zakładki „notatki”. Będzie w niej mógł napisać wiadomość, którą chce przekazać, a następnie wybrać osobę, której chce ją przekazać. Opcjonalnie będzie mógł wybrać czas, w którym notatka pojawi się u wybranego użytkownika oraz czas po jakim zniknie (ponieważ będzie nieaktualna).

5.18.5 Użytkownicy mogący korzystać - menadżer, pracownik

5.18.6 Powiązane wymagania funkcjonalne - brak

5.19.1 Nazwa - Wyświetlanie i przechowywanie numerów telefonów

5.19.2 ID - WiPNT

5.19.3 Cel - Dzięki tej funkcji wszyscy użytkownicy będą mieli dostęp do bazy potrzebnych numerów telefonów. Np. każdy pracownik będzie mógł zadzwonić do serwisu.

5.19.4 Opis działania - Aby zobaczyć spis telefonów będzie trzeba będzie wejść w zakładkę „kontakty”. W tej zakładce zostaną wyświetlone w formie listy wszystkie zapisane numery wraz z ich nazwami. Z tego miejsca będzie można również dodawać kolejne pozycje do listy kontaktów.

5.19.5 Użytkownicy mogący korzystać - menadżer, pracownik

5.19.6 Powiązane wymagania funkcjonalne - brak

5.20.1 Nazwa - Wezwanie ochrony

5.20.2 ID - WO

5.20.3 Cel - Aplikacja zapewni możliwość wezwania ochrony poprzez naciśnięcie prostej kombinacji przycisków. Dzięki temu, gdy nastąpi taka potrzeba pracownik pubu będzie mógł szybko i bez wiedzy osób będących w pubie wezwać ochronę.

5.20.4 Opis działania - Aby wezwać ochronę użytkownik będzie musiał wcisnąć kombinacją przycisków „ctrl” + „alt” + „o”, gdy aplikacja będzie aktywna. Aplikacja automatycznie wyśle wiadomość tekstową firmie ochroniarskiej opłacanej przez pub.

5.20.5 Użytkownicy mogący korzystać - menadżer, pracownik

5.20.6 Powiązane wymagania funkcjonalne - brak

6. Wymagania niefunkcjonalne

6.1 Poniższa część dokumentu zawiera specyfikacje wymagań niefunkcjonalnych tworzonego systemu.

6.2 Użyteczność - Łatwość korzystania z aplikacji, szybkość uczenia się obsługi aplikacji. Zapewniona zostanie poprzez interfejs graficzny GUI stworzony zgodnie z obowiązującymi standardami. W zapewnieniu użyteczności aplikacji pomoże również wbudowany w aplikację samouczek.

6.3 Intuicyjność - Estetyczny i przyjazny interfejs użytkownika.

6.4 Dostępność - Aplikacja po stronie serwera będzie dostępna przez 99.9% czasu działania serwera (awarie serwera mogą uniemożliwić jej działanie).

6.5 Niezawodność - Rozpoczęcie realizacji naprawy wad i błędów nastąpi do tygodnia po ich zgłoszeniu.

6.6 Rozszerzalność - Możliwość rozbudowy aplikacji o nowe funkcje. Dane aplikacji przechowywane będą w bazie danych na serwerze. Zapewni to możliwość stworzenia kolejnych aplikacji na inne systemy operacyjne, z możliwością synchronizowania wszystkich danych.

6.7 Kompatybilność - Kompatybilność z serwerem Tomcat, bazą danych mySql, wersją Javy SE 9 i nowszymi.

6.8 Integralność - Nie będzie możliwe wprowadzenie zmian aplikacji w nieautoryzowany sposób.

6.9 Skalowalność - Zapewnienie stabilnej pracy wraz ze wzrostem ilości danych w aplikacji.

6.10 Bezpieczeństwo (safety) - Utrudniony dostęp do danych dla osób nieupoważnionych.

6.11 Zabezpieczenie (security) - Szyfrowanie hasła w bazie danych, hermetyzacja danych.

6.12 Przenośność - Łatwość zmiany lokalizacji aplikacji, środowiska, możliwy eksport danych.

6.13 Wydajność - Korzystny czas reakcji na zapytania użytkowników, zapewnienie optymalizacji systemu baz danych.

7. Technologie

7.1 Poniższa część dokumentu opisuje jakie technologie zostaną użyte do stworzenia aplikacji.

7.2 Apache Maven - narzędzie automatyzujące budowę oprogramowania na platformę Java. Poszczególne funkcje Mavena realizowane są poprzez wtyczki, które są automatycznie pobierane przy ich pierwszym wykorzystaniu. Plik określający sposób budowy aplikacji nosi nazwę POM-u (ang. *Project Object Model*). Budowanie oprogramowania ma zakończyć się osiągnięciem wybranego przez budującego **celu**. Dostępnych celów jest wiele. Pula celów nie jest bezpośrednio określona przez twórcę POM-a, tak jak ma to miejsce w przypadku Anta lub Make'a, lecz przez wtyczki rozszerzające funkcjonalność Mavena. Poszczególne cele mogą wymagać wcześniejszej realizacji innych celów, np. cel *package* (zbudowanie paczki dystrybucyjnej) wymaga uprzedniej realizacji *compile* (kompilacja kodów źródłowych) oraz *test* (uruchomienie testów automatycznych).

7.3 Wzorzec MVC - Model-View-Controller (pol. *Model-Widok-Kontroler*) wzorzec architektoniczny służący do organizowania struktury aplikacji posiadających graficzne interfejsy użytkownika. Model-View-Controller zakłada podział aplikacji na trzy główne części:

- Model – jest pewną reprezentacją problemu bądź logiki aplikacji.
- Widok – opisuje, jak wyświetlić pewną część modelu w ramach interfejsu użytkownika. Może składać się z podwidoków odpowiedzialnych za mniejsze części interfejsu.
- Kontroler – przyjmuje dane wejściowe od użytkownika i reaguje na jego poczynania, zarządzając aktualizacje modelu oraz odświeżenie widoków.

Wszystkie trzy części są ze sobą wzajemnie połączone.

7.4 IDE IntelliJ - program służących do tworzenia, modyfikowania, testowania i konserwacji oprogramowania. Programy będące zintegrowanymi środowiskami programistycznymi charakteryzują się tym, że udostępniają złożoną, wieloraką funkcjonalność obejmującą edycję kodu źródłowego, kompilowanie kodu źródłowego, tworzenie zasobów programu (tzn. formatek/ekranów/okien dialogowych, menu, raportów, elementów graficznych jak ikony, obrazy), tworzenie baz danych, komponentów i innych.

7.5 Java SE 9, EE 8 - współbieżny, oparty na klasach, obiektowy język programowania ogólnego zastosowania. Java jest językiem tworzenia programów źródłowych kompilowanych do kodu bajtowego, czyli postaci wykonywanej przez maszynę wirtualną. Język cechuje się silnym typowaniem.

7.6 GUI Swing - biblioteka graficzna używana w języku programowania Java. Jest nowszą, ulepszoną wersją biblioteki AWT.

7.7 baza danych MySQL - MySQL rozwijany jest przez firmę Oracle. MySQL był pisany raczej z myślą o szybkości niż kompatybilności ze standardem SQL. MySQL obsługuje większą część obecnego standardu ANSI/ISO SQL (tj. SQL:2003). Wprowadza również swoje rozszerzenia i nowe elementy języka.

7.8 Serwer Tomcat - kontener aplikacji webowych (ang. „web container") rozwijany w ramach projektu Apache. Jako kontener aplikacji jest serwerem, który umożliwia uruchamianie aplikacji internetowych w technologiach Java Servlets i Java Server Pages (JSP). Jest to jeden z bardziej popularnych kontenerów Web. Tomcat jest wykorzystywany w takich serwerach aplikacji JEE (J2EE) jak Apache Geronimo. Jest również bardzo popularnym kontenerem dla samodzielnych aplikacji (niewymagających pełnego serwera aplikacji) pisanych w środowisku Spring Framework.

7.9 Spring Framework - szkielet tworzenia aplikacji (ang. application framework) w języku Java dla platformy Java Platform, Enterprise Edition (aczkolwiek istnieje też wersja dla środowiska .NET). Spring Framework powstał jako alternatywa dla programowania aplikacji z użyciem Enterprise JavaBeans. Programowanie z użyciem EJB narzucało wiele ograniczeń – wymagając między innymi przyjęcia określonego modelu tworzenia oprogramowania. Funkcjonalność EJB okazała się także za "ciężka" do wszystkich zastosowań (w małych projektach wykorzystywano tylko niewielką część oferowanej przez EJB funkcjonalności) jednocześnie stworzenie małej aplikacji w środowisku EJB wymagało nakładu pracy jak przy aplikacji dużej. Odmienna koncepcja Springa – lekkiego szablonu, który nie wymusza specyficznego modelu programowania, stała się bardzo popularna wśród programistów Javy. Spring Framework oferuje dużą swobodę w tworzeniu rozwiązań, a jednocześnie jest dobrze udokumentowany i zawiera rozwiązania wielu zagadnień, często występujących w programowaniu.

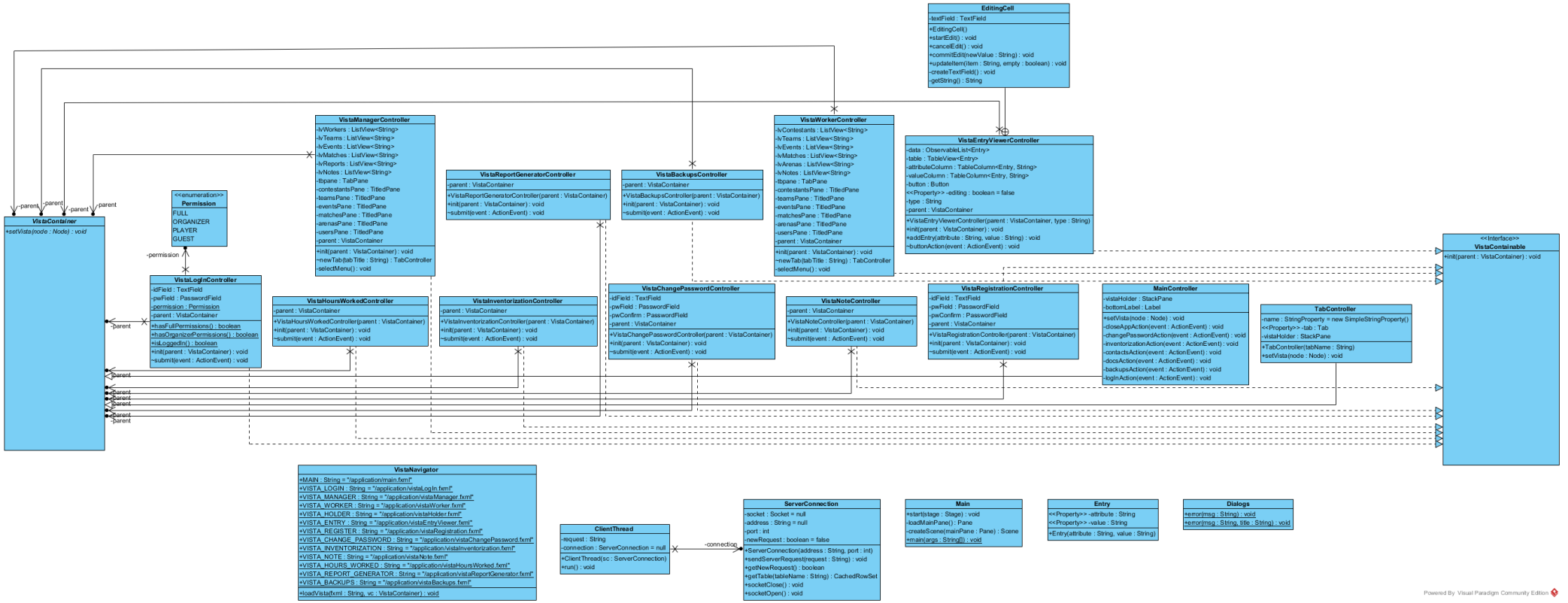
7.10 Hibernate - framework do realizacji warstwy dostępu do danych (ang. persistence layer). Zapewnia on przede wszystkim translację danych pomiędzy relacyjną bazą danych a światem obiektowym (ang. O/R mapping). Opiera się na wykorzystaniu opisu struktury danych za pomocą języka XML, dzięki czemu można rzutować obiekty, stosowane w obiektowych językach programowania, takich jak Java bezpośrednio na istniejące tabele bazy danych. Dodatkowo Hibernate zwiększa wydajność operacji na bazie danych dzięki buforowaniu i minimalizacji liczby przesyłanych zapytań. Jest to projekt rozwijany jako open source.

7.11 System kontroli wersji GitHub - hostingowy serwis internetowy przeznaczony dla projektów programistycznych wykorzystujących system kontroli wersji Git.

7.12 MySQL – wolnodostępny system zarządzania relacyjnymi bazami danych. MySQL rozwijany jest przez firmę Oracle. Wcześniej przez większość czasu jego tworzeniem zajmowała się szwedzka firma MySQL AB. MySQL AB została kupiona 16 stycznia 2008 roku przez Sun Microsystems, a ten 27 stycznia 2010 roku przez Oracle.

Na następnej stronie umieszczono diagram klas.

8. Diagram klas



9. Słownik klas

ClientThread - wątek odpowiedzialny na połączenie z serwerem.

Dialogs - klasa zawierająca statyczne metody pozwalające na wyświetlanie alertów.

Entry - obiekty tej klasy reprezentują wpis w tabeli.

Main - główna klasa programu - otwiera interfejs graficzny i potrzebne wątki.

MainController - klasa kontrolująca główne okno aplikacji.

Permission - uprawnienia występujące w systemie.

ServerConnection - zawiera dane na temat połączenia z serwerem oraz metody umożliwiające komunikację z nim.

TabController - kontroluje okna zakładek.

VistaBackupsController - kontroluje okno kopii zapasowych.

VistaChangePasswordController - kontroluje okno zmiany hasła.

VistaContainer - klasa abstrakcyjna, po której dziedziczą klasy mogące przechowywać widoki (wyświetlać zagnieżdżone okna).

VistaEntryViewerController - kontroler okna wyświetlającego tabele.

VistaInventorizationController - kontroler okna inwentaryzacji.

VistaLogInController - kontroler okna logowania.

VistaManagerController - kontroler okna menadżera.

VistaNavigator - klasa zawierająca statyczne odnośniki pomiędzy oknami/widokami.

VistaNoteController - kontroler okna notatek.

VistaRegistrationController - kontroler okna rejestracji.

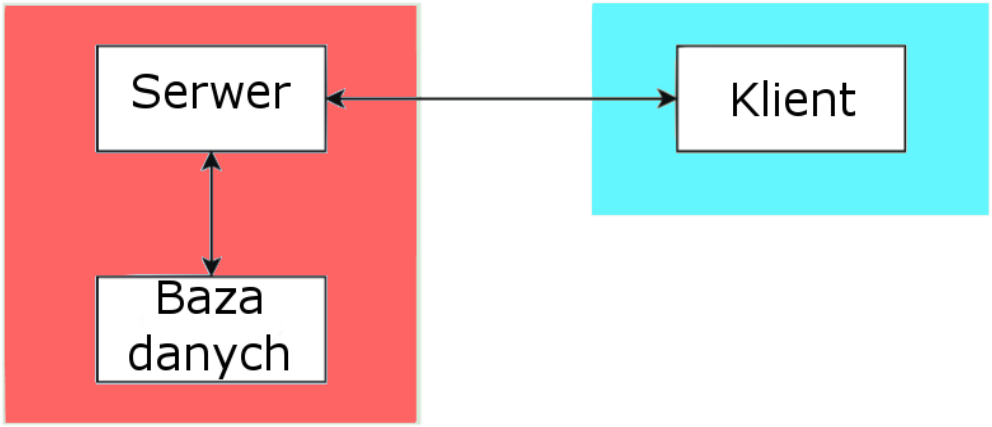
VistaReportGeneratorController - kontroler okna raportów.

VistaWorkerController - kontroler okna pracowników.

10. Architektura systemu

Baza danych zarządzana będzie przez aplikację serwerową obsługującą zapytania od Klienta.

Aplikacja kliencka ma się skupiać na przyjmowaniu poleceń od użytkownika, wysyłaniu ich do serwera i wyświetlaniu wyników po otrzymaniu odpowiedzi.



Architektura typu klient-serwer pozwoli na jednoczesny dostęp wielu użytkowników z różnymi uprawnieniami.

11. Baza danych

Baza danych została wykonana w systemie MySQL ze względu na jego prostotę obsługi oraz popularność, co przekłada się na jego dostępność w serwisach hostingowych. W celu testów optymalizacyjnych baza została wypełniona danymi ze strony filldb.info. Ilość wypełnionych rekordów została dobrana tak, aby zasymulować realne użycie aplikacji.

Zabiegi optymalizacyjne

W celu przyśpieszenia działania bazy danych zostały wykonane następujące zabiegi optymalizacyjne. Profile dotyczą prostych zapytań z użyciem indeksów.

Indeksy unikalne

Nałożono indeksy unikalne na kolumny tam gdzie zakładamy, że nie wystąpią dwa wpisy o takiej samej wartości.

Indeksy nieunikalne

Tam gdzie to uzasadnione, ale wartości występują wartości powtarzające się nałożyliśmy indeksy nieunikalne.

Profil jednego z zapytań przed zastosowaniem indeksów

Szczegółowy profil

Kolejność	Stan	Czas
1	Starting	28 µs
2	Checking Permissions	6 µs
3	Opening Tables	19 µs
4	After Opening Tables	5 µs
5	System Lock	4 µs
6	Table Lock	4 µs
7	Init	15 µs
8	Optimizing	7 µs
9	Statistics	19 µs
10	Preparing	9 µs
11	Executing	3 µs
12	Sending Data	671 µs
13	End	9 µs
14	Query End	4 µs
15	Closing Tables	3 µs
16	Unlocking Tables	8 µs
17	Freeing Items	4 µs
18	Updating Status	29 µs
19	Cleaning Up	4 µs

Podsumowanie według stanu

Stan	Całkowity czas	% Czas	Połączenia	ø Czas
Sending Data	671 µs	78.85%	1	671 µs
Updating Status	29 µs	3.41%	1	29 µs
Starting	28 µs	3.29%	1	28 µs
Opening Tables	19 µs	2.23%	1	19 µs
Statistics	19 µs	2.23%	1	19 µs
Init	15 µs	1.76%	1	15 µs
Preparing	9 µs	1.06%	1	9 µs
End	9 µs	1.06%	1	9 µs
Unlocking Tables	8 µs	0.94%	1	8 µs
Optimizing	7 µs	0.82%	1	7 µs
Checking Permissions	6 µs	0.71%	1	6 µs
After Opening Tables	5 µs	0.59%	1	5 µs
System Lock	4 µs	0.47%	1	4 µs
Table Lock	4 µs	0.47%	1	4 µs
Query End	4 µs	0.47%	1	4 µs
Freeing Items	4 µs	0.47%	1	4 µs
Cleaning Up	4 µs	0.47%	1	4 µs
Executing	3 µs	0.35%	1	3 µs
Closing Tables	3 µs	0.35%	1	3 µs

Profil zapytania po zastosowaniu indeksów

Szczegółowy profil

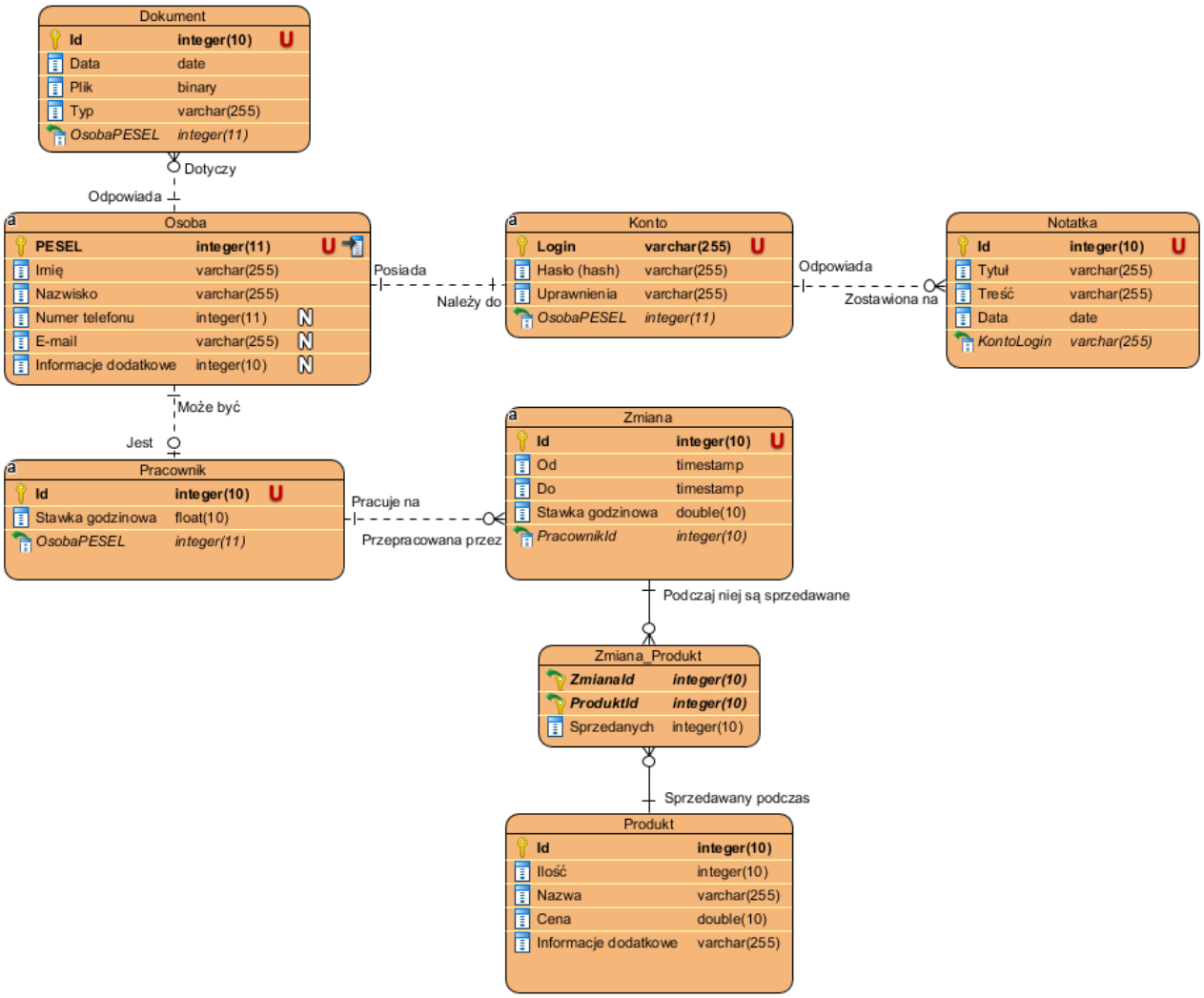
Kolejność	Stan	Czas
1	Starting	40 µs
2	Checking Permissions	10 µs
3	Opening Tables	34 µs
4	After Opening Tables	11 µs
5	System Lock	6 µs
6	Table Lock	21 µs
7	Init	28 µs
8	Optimizing	14 µs
9	Statistics	66 µs
10	Preparing	10 µs
11	Unlocking Tables	13 µs
12	Executing	16 µs
13	Query End	11 µs
14	Closing Tables	5 µs
15	Unlocking Tables	18 µs
16	Freeing Items	9 µs
17	Updating Status	45 µs
18	Cleaning Up	7 µs

Podsumowanie według stanu

Stan	Całkowity czas	% Czas	Połączenia	ø Czas
Statistics	66 µs	18.13%	1	66 µs
Updating Status	45 µs	12.36%	1	45 µs
Starting	40 µs	10.99%	1	40 µs
Opening Tables	34 µs	9.34%	1	34 µs
Init	28 µs	7.69%	1	28 µs
Table Lock	21 µs	5.77%	1	21 µs
Executing	16 µs	4.40%	1	16 µs
Optimizing	14 µs	3.85%	1	14 µs
Unlocking Tables	13 µs	3.57%	1	13 µs
After Opening Tables	11 µs	3.02%	1	11 µs
Query End	11 µs	3.02%	1	11 µs
Checking Permissions	10 µs	2.75%	1	10 µs
Preparing	10 µs	2.75%	1	10 µs
Freeing Items	9 µs	2.47%	1	9 µs
Cleaning Up	7 µs	1.92%	1	7 µs
System Lock	6 µs	1.65%	1	6 µs
Closing Tables	5 µs	1.37%	1	5 µs

Jest widoczna duża poprawa w czasie wykonania zapytania.

12. Schemat bazy danych



13. Źródła

13.1 Poniższa część dokumentu zawiera informację o źródłach pomocnych przy jego tworzeniu.

13.2 Dokumenty powstały w konsultacji z właścicielem Pubu Drewutnia Markiem Walkosz.

13.3 Serwisy użyte przy tworzeniu dokumentu (odnośniki aktualne na dzień 02.12.2018r.):

- http://riad.pk.edu.pl/~drab/studenci.php
- https://pl.wikipedia.org/wiki/GitHub
- https://pl.wikipedia.org/wiki/Hibernate
- https://pl.wikipedia.org/wiki/Spring_Framework
- https://pl.wikipedia.org/wiki/Apache_Tomcat
- https://pl.wikipedia.org/wiki/MySQL
- https://pl.wikipedia.org/wiki/Swing_(Java)
- https://pl.wikipedia.org/wiki/Java
- https://pl.wikipedia.org/wiki/IntelliJ_IDEA
- https://pl.wikipedia.org/wiki/Zintegrowane_%C5%9Brodowisko_programistyczne
- https://pl.wikipedia.org/wiki/Model-View-Controller
- https://pl.wikipedia.org/wiki/Apache_Maven
- https://sjp.pwn.pl/
- https://translate.google.com/?source=gtx
- https://javastart.pl/
- https://pl.wikipedia.org/wiki/MySQL

13.4 Literatura użyta przy tworzeniu dokumentu:

- Java. Efektywne programowanie. Wydanie III - Joshua Bloch
- Bazy danych. Podstawy projektowania i języka SQL (ebook) - Krystyna Czapla
- Docker. Projektowanie i wdrażanie aplikacji - Jarosław Krochmalski