**Baza danych dla ewidencji magazynowej**

1. **Cel zadania**

Celem zadania jest realizacja systemu bazodanowego, pozwalającego na kompleksową obsługę magazynów. System powinien składać się z aplikacji klienckiej oraz bazy danych MySQL.

1. **Słowny opis projektu**
   1. **Aplikacja kliencka**

Aplikacja kliencka powinna umożliwiać pracownikom dodawanie, edycje usuwanie oraz realizacje zamówień. Ważne jest także aby umożliwiała dostęp do historii zamówień, ewidencji klientów czy dostawców, a także łatwe operowanie na zgromadzonych w bazie danych. Możliwe musi również być załączanie faktury czy tworzenie korekt.

* 1. **Serwer oraz baza danych MySQL**

Kontener danych zrealizowany zostanie za pomocą darmowego MySQL umieszczonego na darmowym serwerze, prawdopodobnie Apache. Musi umożliwiać realizację wymagań postawionych przez użytkownika i funkcji udostępnianych przez aplikację kliencką.

1. **Wymagania funkcjonalne**

|  |  |  |  |
| --- | --- | --- | --- |
| Id | **Nazwa** | **Opis** | **Priorytet** |
| 001 | Obsługa zamówień zewnętrznych | System pozwala na przyjmowanie oraz obsługę zamówień od klientów | Wysoki |
| 002 | Obsługa zamówień wewnętrznych | System pozwala na przyjmowanie oraz obsługę zamówień od innych magazynów | Wysoki |
| 003 | Ewidencja towarów | System pozwala na ewidencje towarów | Wysoki |
| 004 | Podział na kategorie | System pozwala na podział towarów na kategorie | Średni |
| 005 | Przechowywanie informacji o pracownikach, klientach i dostawcach | System pozwala na przechowywanie informacji o pracownikach, klientach i dostawcach | Średni |
| 006 | Przeglądanie historii zamówień | System pozwala na przeglądanie historii zamówień | Średni |
| 007 | Ewidencja faktur | System pozwala na ewidencje faktur | Wysoki |

1. **Wymagania niefunkcjonalne**

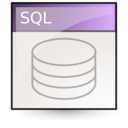
* System współpracuje z serwerem Apache oraz bazą danych MySQL
* System jest kompatybilny z systemami operacyjnymi Windows XP, Vista, 7 w wersjach 32 bitowych
* System powinien obsługiwać do 500 użytkowników jednocześnie.

1. **Diagram ER bazy danych oraz propozycje indeksów**

[](model.pdf)

*model.pdf*

1. **Skrypt DDL**

[](db_create.sql)

*db\_create.sql*

1. **Hierarchia menu**

Menu aplikacji składać się będzie z następujących podmenu:

* Plik:
  + Zapisz parametry połączenia
  + Wczytaj parametry połączenia
  + Zapisz widok do pliku HTML
  + Zakończ

Podstawową funkcjonalnością menu plik jest możliwość wyeksportowania parametrów połączenia do pliku oraz zapis aktualnie widocznej tabeli jako HTML.

* Baza danych:
  + Ustaw dane połączenia
  + Połącz
  + Rozłącz

Aby umożliwić użytkownikowi zmianę parametrów połączenia z bazą danych w prosty sposób.

* Zarządzaj:
  + Zamówienia wejściowe
  + Zamówienia wyjściowe
  + Kategorie
  + Części
  + Lokalizacje
  + Osoby:
    - Pracownicy
    - Klienci
    - Dostawcy

Jest to kompleksowe menu służące do realizacji podstawowych funkcjonalności programu. Każda z opcji wyświetla na ekranie odpowiedni ekran, charakterystyczne dla siebie GUI, umożliwiające dodanie, usunięcie oraz modyfikację poszczególnych elementów.

* Raporty:
  + Zamówienia zrealizowane
  + Zamówienia oczekujące na realizację
  + Faktura zamówienia
  + Wykaz dostawców
  + Obroty dla lokalizacji

Podmenu raportowe zawiera podstawowe i najczęściej wykorzystywane raporty dzienne i miesięczne. W zależności od wymagań użytkownika istnieje możliwość prostego dodawania nowych raportów do menu.

1. **Podział realizacji funkcjonalności serwera i klienta**

**Klient**

- wyświetlanie żądanych danych z bazy danych  
- obsługa edycji, dodawania i usuwania rekordów  
- udostępnianie przyjaznego interfejsu użytkownik   
- możliwie dokładna wstępna walidacja wprowadzonych danych

**Serwer**

- końcowa walidacja danych  
- logowanie zdarzeń i akcji  
- obsługa sesji  
- pobieranie, edycja oraz usuwanie rekordów  
- umożliwienie nawiązania wielu połączeń jednocześnie  
- realizacja powyższych za pomocą procedur wbudowanych i triggerów

1. **Integralność danych**

Integralność danych na etapie dodawania rekordów zapewniona jest poprzez odpowiednio zaplanowane relacje między tabelami. Nie jest na przykład możliwa sytuacja, w której dwie faktury przypisane są do jednego zamówienia.

W procesie usuwania, z bazy danych muszą zniknąć wszystkie rekordy powiązane z tym usuwanym. Np. jeśli rozwiązujemy kontakt z danym dostawcą, musimy zadbać także o anulowanie wszystkich zamówień z nim powiązanych.

1. **Szkielet aplikacji – podział na moduły**

[](moduly.png)

*moduly.png*

Moduły przedstawione w pliku komunikują się także między sobą.

1. **Opis transakcji dla modułu zamówień**

Transakcja oznacza zablokowanie bazy danych przed reagowaniem na kolejne zapytania. Te są buforowane, a następnie wykonane zbiorowo. W przypadku niepowodzenia w wykonywaniu choć jednego z zapytań, przywracany jest stan sprzed rozpoczęcia transakcji.

W naszym module konieczna będzie transakcyjność w następujących przypadkach:

* *Złożenie zamówienia* – nie powinno być możliwe złożenie nowego zamówienia, jeśli z jakiegoś powodu nie uda się wygenerować i przypisać do niego nowej faktury lub zebrać wszystkich pozycji.
* *Usunięcie zamówienia –* należy zadbać o to, aby przy usuwaniu zamówienia zniknęły także jego pozycje, a także wygenerowana faktura. Nie powinno być w systemie faktur, dla których nie istnieją szczegóły zamówienia.

1. **Projekt GUI dla modułu zamówień**