STUDENCKA PRACOWNIA BAZ DANYCH

Instytut Informatyki Uniwersytetu Wrocławskiego

Łukasz Czapliński

Dokumentacja projektu

Coffee Shop

Dokumentacja ogólna Wersja 1.0

Tabela 1. Historia zmian

Wersja Da	ta	Opis			Autor		
1.0 20	14-06-13	Powstanie dokumentu			Łukasz Czapliński		
Spis treści							
1. Wprowadzenie							
1.1. Cel dokumentu							
2. Zadania projektu 3						. 3	
3. Model						3	
3.1. Mod	el koncept	tualny				3	
3.2. Mod	el fizyczny	<i>y</i>				3	
3.2.1	. Dostav	vca				3	
3.2.2	. Klient					3	
3.2.3	. Właści	ciel				4	
3.2.4	. Typ pr	oduktu				4	
3.2.5	. Produl	kt				4	
3.2.6	. Zamów	vienie				4	

1. Wprowadzenie

1.1. Cel dokumentu

Dokument ten ma na celu sprecyzowanie zadań projektu CoffeeShop i sposobów ich realizacji.

2. Zadania projektu

Celem projektu jest stworzenie aplikacji bazodanowej pozwalającej na obsługę prostego CoffeeShopu. Ma być on zdecentralizowany – kilku właścicieli, z których każdy ma swoje towary. Mogą oni współdzielić dostawców towarów. Klienci mają mieć wybór co kupują od którego właściciela. Powinno się to odbywać zdalnie – zarówno pomiędzy klientami a właścicielami (klient zamawia towary u konkretnego dostawcy, a ten kontaktuje się z nim w sprawie odbioru i zapłaty) oraz dostawcami i właścicielami (właściciel wybiera jaki typ produktu chce zamówić i kontaktuje się z dostawcą lub odwrotnie – dostawca pyta właścicieli z którymi współpracował czego będą potrzebować).

3. Model

3.1. Model konceptualny

Jest przedstawiony na rys. [1].

Można w nim wyróżnić 3 główne role: klient, dostawca i właściciel oraz 3 obiekty którymi operują: produkty, ich typy oraz zamówienia.

Klient składa zamówienia do właściciela na konkretne produkty.

Właściciel dodaje nowe produkty wybierając z oferty dostawców.

Dostawca rejestruje jakie typy produktów w jakiej cenie może zapewnić.

Zamawianie dostaw nie jest kontrolowane przez tą bazę danych – odbywa się pomiędzy właścicielem a dostawcą przez umówione kontakty – maile.

3.2. Model fizyczny

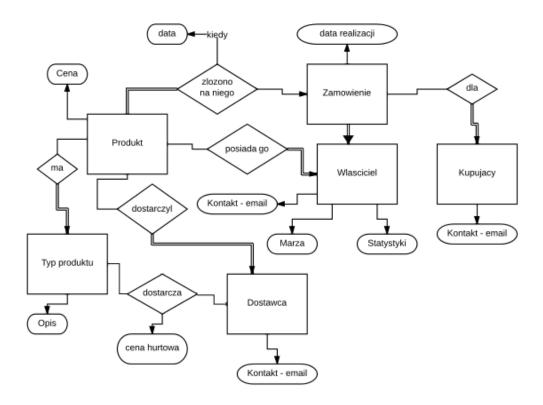
3.2.1. Dostawca

```
create sequence doid_seq;
create table dostawca (
   doid integer primary key default nextval('doid_seq'),
   mail text unique check (mail ~ '(([a-z]|[._,-]|\d)+)@(([a-z]|[_,-]|
);
```

3.2.2. Klient

```
drop table if exists kupujacy cascade;
drop sequence if exists kuid_seq cascade;
create sequence kuid_seq;
```

```
create table kupujacy (
  kuid integer primary key default nextval('kuid_seq'),
  mail text unique check (mail ~ '(([a-z]|[._,-]|\d)+)@(([a-z]|[_,-]|
);
3.2.3. Właściciel
create sequence wlid_seq;
create table wlasciciel (
  wlid integer primary key default nextval('wlid_seq'),
  mail \ text \ \textbf{unique check} \ (\, mail \ \sim \ \, '\,(\,(\,[\,a-z\,\,]\,|\,[\,.\,\_,\,-\,]\,|\,\backslash\,d\,) + ) @\,(\,(\,[\,a-z\,\,]\,|\,[\,\_,\,-\,]\,|\,\backslash\,d) + ) \\
  marza \ decimal(3,2) \ not \ null \ check \ ( marza < 1 \ and \ marza > 0)
);
3.2.4. Typ produktu
create sequence tpid_seq;
create table typ_produktu (
  tpid integer primary key default nextval('tpid_seq'),
  nazwa text,
  opis text
);
3.2.5. Produkt
create sequence prid_seq;
create table produkt (
  prid integer primary key default nextval('prid_seq'),
  tpid integer not null references typ produktu,
  cena decimal (6,2) not null,
  zaid integer null references zamowienie,
  doid integer not null references dostawca,
  wlid integer not null references wlasciciel
);
3.2.6. Zamówienie
create sequence zaid_seq;
create table zamowienie (
  zaid integer primary key default nextval ('zaid_seq'),
  realizacja date null check (realizacja >= zlozenie),
  zlozenie date null,
  wartosc integer not null default 0,
  kuid integer not null references kupujacy,
  wlid integer not null references wlasciciel
);
```



Rysunek 1. Model konceptualny bazy danych