Національний лісотехнічний університет України
(повне найменування вищого навчального закладу)
Навчально – науковий інститут комп'ютерних наук та інформаційних технологій
(повне найменування інституту, назва факультету (відділення))

Кафедра комп'ютерних наук
(повна назва кафедри (предметної, циклової комісії))

Пояснювальна записка

до дипломної роботи
перший (бакалаврський)
(рівень вищої освіти)

на тему: **Розроблення інформаційної системи аналізу функціонування мережі магазинів спортивних товарів**

Виконав: студент IV курсу, групи **КН–41**
спеціальності 122 – "Комп'ютерні науки"
(шифр і назва напряму підготовки, спеціальності)
Зеліско Т.М.
(прізвище та ініціали)

Керівник **Думанський О.І.**
(прізвище та ініціали)

Рецензент **Карашецький В.П.**
(прізвище та ініціали)

Львів – 2025 р.

Національний лісотехнічний університет України
(повне найменування вищого навчального закладу)

ННІкомп'ютернихнаук та інформаційних технологій
Кафедра комп'ютернихнаук
Рівень вищої освітиперший (бакалаврський)
Спеціальність122 – "Комп'ютерні науки"
(шифр і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри КН
_____ Борецька І.Б.
"10" червня 2025 року

З А В Д А Н Н Я
на дипломну роботу студенту

Зеліску Тарасу Миколайовичу
(прізвище, ім'я, по батькові)

1.Тема роботи"*Розроблення інформаційної системи аналізу функціонування мережі магазинів спортивних товарів*"
Керівник роботи *Думанський Остап Іванович, канд. фіз.–матем. наук, доцент*
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджений наказом вищого навчального закладу від "15" листопада 2024 року № С-882
2.Термін подання студентом (роботи)10.06.2025 р.
3.Вихідні дані до роботи (проєкту). *Аналіз шляхів вирішення поставлених задач, організаційна структура застосунку , огляд алгоритмів та програмних засобів для їх реалізації.*
4.Зміст пояснювальної записки (перелік питань, які потрібно розробити)
Вступ.
Розділ 1.Стан проблемної області.
Розділ 2. Інформаційне та математичне забезпеченн.
Розділ 3. Програмне та технічне забезпечення
Висновки. Список використаних літературних джерел.
5.Перелік графічного матеріалу: *слайди для доповіді.*
6.Дата видачі завдання 18листопада 2024р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Термін виконання етапів проекту (роботи)	Примітка
1	Огляд літературних та інших джерел згідно досліджуваної теми.	18.11.2024 р. – 20.12.2024р.	*Виконано*
2.	Аналіз досліджуваної теми та вибір відповідних варіантів її розроблення	10.01.2025р. – 25.01.2025р.	*Виконано*
3.	Постановка задачі та її формалізація	01.02.2025р.	*Виконано*
4.	Вибір та обґрунтування методів і засобів проведення дослідження	10.02.2025р. – 20.02.2025 р.	*Виконано*
5.	Розроблення концептуальної схеми БД та структури таблиць	05.03.2025р. – 15.03.2025 р.	*Виконано*
6.	Програмна реалізація завдання	20.03.2025р. – 05.04.2025 р.	*Виконано*
7.	Тестування програмного продукту та отриманих результатів	10.04.2024р. – 20.04.2024 р.	*Виконано*
8.	Розроблення пояснювальної записки дипломної роботи	01.05.2025р. – 20.05.2025 р.	*Виконано*
9.	Корегування пояснювальної записки згідно вимог, розроблення презентації	02.06.2025р. – 08.06.2025 р.	*Виконано*

Студент _____ **Зеліско Т.М.**
(підпис) (прізвище та ініціали)

Керівник роботи _____ **Думанський О.І.**
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

В даній дипломній роботі представлено інформаційну систему аналізу функціонування мережі магазинів спортивних товарів, яка повинна сприяти полегшенню роботи працівників.

Даний програмний продукт реалізований за допомогою таких мов програмування як: PHP, Jquery, Ajax, Java(JavaScript), MySQL, UIkit.

В дипломній роботі: 51 стор. тексту пояснювальної записки, 8 таблиць, 12 рисунків, 18 посилань.

Ключові слова: продажа товарів, програма , система управління, інформаційна система.

ABSTRACT

In this thesis presented information system analysis of the network of sporting goods stores, which should contribute to facilitating the work of employees.

This software is implemented using programming languages such as: PHP, Jquery, Ajax, Java (JavaScript), MySQL, UIkit.

In the thesis: 51 pages of explanatory note text, 8 tables, 12 figures, 18 references.

Keywords: selling goods, programs, control, information system.

ТЕХНІЧНЕ ЗАВДАННЯ

Необхідно розробити інформаційну система, яка буде надавати послуги працівникам мережі спортивних магазинів. Працівники за допомогою графічного інтерфейсу вводять всю необхідну інформацію в базу даних своєї установи і мають можливість редагувати або видаляти вже введену раніше інформацію.

У розробленій інформаційній системі вводиться наступна інформація: про товар; про категорії товарів; про виробників товарів; про підрозділи мережі магазинів; про надходження товарів на склад; про відправку товару зі складу в підрозділ мережі; про факт продажу товару.

Система повинна давати можливість відстежувати всі рухи товару в мережі магазинів від надходження на склад до продажу, тобто розробити інформаційну систему, яка повинна включати:

- побудову дерево цілей для створення консолідованого інформаційного ресурс;
- проведення вибору та обґрунтування методів реалізації поставленого завдання;
- розроблення зручного інтерфейсу для проведення відповідних етапів функціонування інтернет-магазину.

На основі результатів дослідження розробити відповідні рекомендації управління інформаційними потоками у бізнесі віртуальної торгівлі.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАЧЕНЬ

HTML – мова логічної розмітки тексту;

CSS – каскадна таблиця стилів;

ІС – інформаційна система;

СУБД – система управління базами даних.

PHP – препроцессор гипертексту;

MySQL – вільна реляційна система управління базами даних;

БД – База Даних;

Jquery – бібліотека JavaScript;

JavaScript – прототипно-орієнтована мова програмування;

API (Application Programming Interface) – прикладний програмний інтерфейс;

ВСТУП

У сучасному суспільстві кожна стабільна компанія повинна мати власний сайт в мережі Інтернет, який забезпечить інформаційну підтримку існуючого бізнесу. Проект реалізовано використовуючи технології MySQL, HTML, CSS, PHP та JavaScript [12, 19, 20].

Розробка сайтів для компаній є актуальною і затребуваною сферою діяльності, тому що сайт фірми в мережі Інтернет являє собою досить дешевий і масовий спосіб реклами, дає можливість потенційним та існуючим клієнтам легко отримувати інформацію про товари і послуги компанії, її ділових інтересах, що може допомогти знайти нових замовників і партнерів по бізнесу, а, отже, сприяє збільшенню обсягу продажів і рентабельності підприємства.

Об'єктом дослідження даного проекту є аналіз процесу електронної торгівлі. Мережа магазинів займається продажами спортивних товарів. Система повинна давати можливість відстежувати всі рухи товару в мережі магазинів від надходження на склад до продажу.

Мета кваліфікаційної роботи - розробка веб-інформаційної системи електронної комерції.

У дипломній роботі:

- Проведено дослідження предметної області - сфери послуг;
- Зроблений огляд найбільш відомих існуючих інформаційних систем у сфері діяльності мережі спортивних магазинів;
- Розроблена веб-інформаційна система;
- Проведений аналіз технологій і засобів розробки інформаційної системи.

Призначення системи. Розроблена інформаційна система буде надавати послуги працівникам мережі спортивних магазинів. Працівники за допомогою графічного інтерфейсу вводять всю необхідну інформацію в базу даних своєї установи і мають можливість редагувати або видаляти вже введену раніше інформацію.[1,2, 5,7].

В інформаційній системі вводиться наступна інформація:

- про товар та його категорію;
- про виробників товарів;
- про підрозділи мережі магазинів;
- про надходження товарів на склад, відправку товару зі складу в підрозділ мережі та факт продажу товару.

Розділ 1. СТАН ПРОБЛЕМНОЇ ОБЛАСТІ

1.1. Поняття інформаційної системи

Інформаці́йна систе́ма (англ. Information system) — сукупність організаційних і технічних засобів для збереження та обробки інформації з метою забезпечення інформаційних потреб користувачів.

Таке визначення може бути задовільним тільки при найбільш узагальненій і неформальній точці зору і підлягає подальшому уточненню. Інформаційні системи діють в Україні під назвою «автоматизовані системи (АС)».

Історія

Інформаційні системи здавна знаходять (в тому чи іншому вигляді) досить широке застосування в життєдіяльності людства. Це пов'язано з тим, що для існування цивілізації необхідний обмін інформацією — передача знань, як між окремими членами і колективами суспільства, так і між різними поколіннями.

Інформаційні системи існують з моменту появи суспільства, оскільки на кожній стадії його розвитку існує потреба в управлінні. Місією інформаційної системи є виробництво потрібної для організації інформації, потрібної для ефективного управління всіма її ресурсами, створення інформаційного та технічного середовища для управління її діяльністю.

Інформаційна система може існувати і без застосування комп’ютерної техніки — це питання економічної необхідності [4,5,6].

В будь-якій інформаційній системі управління вирішуються задачі трьох типів:

- задачі оцінки ситуації (деколи їх називають задачами розпізнавання образів);
- задачі перетворення опису ситуації (розрахункові задачі, задачі моделювання);
- задачі прийняття рішень (в тому числі і оптимізаційні).

Найдавнішими і найпоширенішими ІС слід вважати бібліотеки. І, дійсно, здавна в бібліотеках збирають книжки (або їх аналоги), зберігають їх, дотримуючись певних правил, створюють каталоги різного призначення для полегшення доступу до книжкового фонду. Видаються спеціальні журнали та довідники, що інформують про нові надходження, ведеться облік видачі.

Найстаріші (у моральному і у фізичному розумінні) ІС повністю базувалися на ручній праці. Пізніше їм на зміну прийшли різні механічні пристрої для обробки даних (наприклад, для сортування, копіювання, асоціативного пошуку тощо). Наступним кроком стало впровадження автоматизованих інформаційних систем (АІС), тобто систем, де для забезпечення інформаційних потреб користувачів використовується ЕОМ зі своїми носіями інформації. В наш час — епоху інформаційної революції — розробляється і впроваджується велика кількість самих різноманітних АІСів з дуже широким спектром використання.

Структура ІС.

Інформаційні системи включають в себе: технічні засоби обробки даних, програмне забезпечення і відповідний персонал. [13, 15, 16]. Чотири складові частини утворюють внутрішню інформаційну основу:

- засоби фіксації і збору інформації;
- засоби передачі відповідних даних та повідомлень;
- засоби збереження інформації;
- засоби аналізу, обробки і представлення інформації.

Класифікація.

1.За ступенем автоматизації

В залежності від ступеня (рівня) автоматизації виділяють ручні, автоматизовані й автоматичні інформаційні системи [1,3, 6,7].

- Ручні ІС

характеризуються тим, що всі операції з переробки інформації виконуються людиною.

- Автоматизовані ІС

частина функції (підсистем) керування або опрацювання даних здійснюється автоматично, а частина — людиною.

- Автоматичні ІС

усі функції керування й опрацювання даних здійснюються технічними засобами без участі людини (наприклад, автоматичне керування технологічними процесами).

2.За сферою призначення

Оскільки ІС утворюються для задоволення інформаційних потреб в межах конкретної предметної галузі, то кожна предметна галузь (в сфері призначення) відповідає свій тип ІС [12, 14, 16]. Перераховувати всі ці типи немає змісту, оскільки кількість предметних галузей велика, але можна вказати наприклад такі типи ІС:

- Економічна ІС — інформаційна система призначена для виконання функцій управління на підприємстві;
- Медична ІС — інформаційна система призначена для використання в лікувальному або лікувально-профілактичному закладі;
- Географічна ІС — інформаційна система, забезпечуюча збір, збереження, обробку, доступ, відображення і розповсюдження даних;
- Адміністративні;
- Виробничі;
- Навчальні;
- Екологічні;
- Юридичні;
- Військові та інші.

1. За місцем діяльності ІС

Класифікація інформаційних систем за місцем діяльності:

- наукові ІС — призначені для автоматизації діяльності науковців, аналізу статистичної інформації, керування експериментом.

- ІС автоматизованого керування — призначені для автоматизації праці інженерів-проектувальників і розроблювачів нової техніки (технології). Такі ІС допомагають здійснювати:
  - ✓ розробку нових виробів і технологій їхнього виробництва;
  - ✓ різноманітні інженерні розрахунки (визначення технічних параметрів виробів, видаткових норм — трудових, матеріальних і т. д.);
  - ✓ створення графічної документації (креслень, схем, планувань);
  - ✓ моделювання проектованих об'єктів;
  - ✓ створення керуючих програм для верстатів із числовим програмним керуванням.
- ІС організаційного керування — призначені для автоматизації функції адміністративного (управлінського) персоналу. До цього класу відносяться ІС керування як промисловими (підприємства), так і непромисловими об'єктами (банки, біржа, страхові компанії, готелі і т. д.) і окремими офісами (офісні системи).
- ІС керування технологічними процесами — призначені для автоматизації різноманітних технологічних процесів (гнучкі виробничі процеси, металургія, енергетикатощо).

Інформаційна система, як система управління, тісно пов’язується, як з системами збереження та видачі інформації, так і з іншої — з системами, що забезпечують обмін інформацією в процесі управління. Вона охоплює сукупність засобів та методів, що дозволяють користувачу збирати, зберігати, передавати і обробляти відібрану інформацію.

4. За функціональним призначенням

В залежності від функціонального призначення можна виділити такі системи:

- Керувальні (АСКТП, АСКВ);
- Проектувальні (САП);
- Наукового пошуку (АСНД, експертні системи);
- Діагностичні, моделювальні;
- Систем підготовки прийняття рішення (СППР).

Типи взаємодії інформаційних систем.

Довільна взаємодія між двома окремими комп'ютерами, наприклад по модему. Обов'язкова участь оператора на приймаючої і передавальної стороні. Можливий обмін в довільному, але заздалегідь обумовленому форматі;

Інтерактивне віддалене взаємодія комп'ютера з інформаційною системою, наприклад по протоколу http. Оператор на передавальній стороні. Як правило використовується певна форма HTML документа. Прийняті документи обробляються автоматично;

Контрольована потокова обробка, наприклад прийом з e-mail, файл містить HTML форму, запуск якої ініціює процес обробки документа або прийом оператором по e-mail електронних документів в обумовленому форматі і далі запуск запуск програми обробки. Вимагає обов'язковий контроль оператора на прийнятої стороні;

Повністю автоматизований процес прийому та обробки електронних документів в обумовленому форматі. Участь операторів не потрібно.

Фактори, що обумовлюють впровадження інформаційних систем.

Основними факторами, які впливають на впровадження інформаційних систем, є потреби організацій та користувачів, а також наявність відповідних засобів для їх формування. Найсуттєвіше на розвиток інформаційних систем вплинули досягнення в галузі комп’ютерної техніки та телекомунікаційних мереж.

1.2 Поняття та основні напрямки електронної комерції

Часто відбувається плутанина двох базових понять: е-комерції та е-бізнесу. Згідно з визначенням фахівців компанії IBM, електронний бізнес - перетворення основних бізнес-процесів за допомогою Інтернет технологій. [1 – 7], Таким чином, електронним бізнесом називають будь-яку ділову активність, що використовує можливості глобальних інформаційних мереж для перетворення внутрішніх і зовнішніх зв'язків з метою створення прибутку.

Електронна комерція є найважливішим складовим елементом електронного бізнесу. Під електронною комерцією маються на увазі будь-які

форми ділової угоди, яка проводиться за допомогою інформаційних мереж. Електронна комерція - це прискорення більшості бізнес-процесів за рахунок їх проведення електронним чином. У цьому випадку інформація передається прямо до одержувача, минаючи стадію створення паперової копії на кожному етапі.

Переваги електронної комерції очевидні, ось лише деякі з них:

• значно збільшується оперативність отримання інформації, особливо при міжнародних операціях;

• значно скорочується цикл виробництва та продажу, тому що більше немає необхідності щоразу вводити отримані документи, до того ж знижується ймовірність виникнення помилок введення;

• значно знижуються витрати, пов'язані з обміном інформацією за рахунок використання дешевших засобів комунікацій;

• використання інтернет - технологій електронної комерції дозволяє компанії стати більш відкритою по відношенню до клієнтів;

• дозволяє легко і швидко інформувати партнерів і клієнтів про продукти та послуги;

• дозволяє створювати альтернативні канали продажів, наприклад, через електронний магазин на корпоративному сайті.

Прийнято виділяти чотири напрями електронної комерції:

- бізнес - бізнес (business-to-business, B2B);
- бізнес - споживач (business-to-consumer, B2C);
- бізнес - адміністрація (business-to-administration, B2A);
- споживач - адміністрація (consumer-to-administration, C2A).

Напрямок бізнес-бізнес включає в себе всі рівні інформаційної взаємодії між компаніями. Вигоди від подібної співпраці важко переоцінити. Наприклад, дилер отримує можливість самостійно розміщувати замовлення і стежити за ходом їх виконання, працюючи з базами даних постачальника і, таким чином, отримуючи необхідну інформацію про запаси продукції на складах. Так само і постачальник, маючи підключення до складських баз, може оперативно

відслідковувати запаси партнера, своєчасно їх поповнюючи. І подібні приклади можна знайти в будь-якій сфері взаємодії між компаніями.

Напрямок бізнес-споживач представляється найбільш перспективним з комерційної точки зору. Його основу складає електронна роздрібна торгівля. В Інтернет працює велике число електронних магазинів, що пропонують широкий спектр товарів і послуг.

Бізнес - адміністрація. Взаємодія бізнесу і адміністрації включає ділові зв'язки комерційних структур з урядовими організаціями, починаючи від місцевої влади і закінчуючи міжнародними організаціям.

Споживач-адміністрація. Цей напрямок найменш розвинений, однак має дуже високий потенціал, який може бути використаний для організації взаємодії уряду і споживача, особливо в соціальній і податковій сфері.

Залежно від ринкової стратегії компанії можливі наступні форми присутності в Інтернет:

• електронна візитна картка;

• електронний каталог;

• електронний магазин.

• торгові інтернет – системи

Електронна візитна картка являє собою кілька сторінок з інформацією про компанію та її діяльність. Основна функція подібного сайту - надати можливість потенційному клієнту познайомитися з послугами компанії, аналогічно звичайному бізнес-довідником або рекламному оголошенню.

Більш просунутою формою інформування клієнта є електронний каталог з докладною інформацією про товари і послуги, а часто - і з поточними цінами.

Електронний магазин дозволяє не тільки вибрати товар або послуги, а й оформити замовлення і зробити покупку через Інтернет.[12,13 – 16]

Нарешті торгова інтернет-система об'єднує в одне ціле інтернет - магазин і традиційний магазин, із загальною системою логістики, управління товарними запасами і т. п.

Платіжна система Інтернет - система проведення розрахунків між фінансовими, бізнес - організаціями та Інтернет - користувачами в процесі купівлі / продажу товарів і послуг через Інтернет.

Саме платіжна система дозволяє перетворити службу по обробці замовлень або електронну вітрину на повноцінний магазин з усіма стандартними атрибутами: вибравши товар або послугу на сайті продавця, покупець може здійснити платіж, не відходячи від комп'ютера.

1.3. Опис веб-орієнтованих інформаційних систем

Успішна діяльність сучасної компанії неможлива без застосування інформаційних систем [13, 14].

У загальному випадку перевагами веб-орієнтованих інформаційних систем є:

- низька сукупна вартість володіння;
- робота з системою можлива не тільки з офісу, а й з будь-якого місця, де є підключення до Інтернету;
- можливість отримати доступ до інформації і працювати з нею з мобільних пристроїв;
- для роботи з системою необхідний тільки інтернет-браузер.

Інформаційні Web-системи отримують особливу перевагу там, де є розподіленість мережі користувачів системи. Це може бути філіальна структура організації, наявність віддалених співробітників або партнерів.

Так само Web-система виявляється більш вигідною у разі потреби мобільного доступу, наприклад коли користувач працює з різних ПК, або йому необхідно мати доступ до системи зі свого портативного комп'ютера, але не бути територіально прив'язаним. У даному випадку вигода полягає у відсутності необхідності установки клієнтського програмного забезпечення та організації доступу до системи (за винятком доступу до інтернету).

Якщо Ви стоїте перед вибором, чи підходить інформаційна система, заснована на технологіях Web, саме Вам, Ви порівняєте свої вимоги зі списком вимог яким Web технології задовольняють найкраще:

- потрібно створити єдиний інформаційний простір для віддалених офісів компанії, клієнтів або партнерів (облікові системи, складські системи, електронні вітрини, інформаційні портали);
- робочі місця користувачів оснащені различающимися платформами (Linux, MacOS, Windows), або їх апаратна конфігурація не дозволяє запуск повновагих додатків (netbook, тонкий клієнт);
- відсутність необхідності в установці і супроводі клієнтських додатків, а також витрат на покупку ліцензій для програмного забезпечення, яке б здійснювало підтримку роботи front-end додатків.

Однак, майте на увазі, що є ряд вимог, які незалежно від платформи роблять реалізацію проекту на основі Web-технологій абсолютно неприйнятною:

- наявність високоінтерактівного інтерфейсу (інтерактивні ігри);
- необхідність роботи на стороні клієнта з додатковим обладнанням (пряме отримання зображень з цифрової камери);
- обробка даних без завантаження на сервер;
- реалізація елементів інтерфейсу поза вікна браузера (іконка в системному треї);
- робота з системою у відсутності підключення до сервера.

Сучасна тенденція розвитку програмних систем така, що частка мережевих віддалених, в тому числі і Web-орієнтованих рішень постійно збільшується. Це зумовлено насамперед меншою вартістю експлуатації таких систем, поліпшенням якості каналів зв'язку і зниженням їх вартості.

Web-технології так само не стояли на місці і набули більш багаті можливості побудови інтерактивного інтерфейсу, сучасні середовища швидкої розробки Web-додатків, можливість взаємодіяти з уже існуючими системами і сервісами [13,16,18, 20].

Основні переваги використання веб-системи для споживача:

1. Відсутність необхідності установки ПЗ на робочих місцях користувачів - доступ до ПЗ здійснюється через звичайний браузер.

2. Радикальне скорочення витрат на розгортання системи в організації. Це витрати на оренду приміщення, організацію дата-центру, оплату праці співробітників і т. д.

3. Скорочення витрат на технічну підтримку і оновлення розгорнутих систем (аж до їх повної відсутності).

4. Швидкість впровадження, обумовлена відсутністю витрат часу на розгортання системи.

5. Зрозумілий інтерфейс - більшість співробітників уже звикли до використання веб-сервісів.

6. Ясність і передбачуваність платежів, захист інвестицій.

7. Можливість отримати більш високий рівень обслуговування ПЗ.

8. Оптимізація витрат на утримання IT фахівців або системних адміністраторів.

1.4. Постановка завдання

У дипломній роботі:

- ✓ Проведести дослідження предметної області - сфери послуг;
- ✓ Зробити огляд найбільш відомих існуючих інформаційних систем;
- ✓ Розробити веб-інформаційну систему;
- ✓ Провести аналіз технологій і засобів розробки інформаційної системи.

Розроблена інформаційна система буде надавати послуги працівникам мережі спортивних магазинів. Працівники за допомогою графічного інтерфейсу вводять всю необхідну інформацію в базу даних своєї установи і мають можливість редагувати або видаляти вже введену раніше інформацію.

В інформаційній системі вводиться наступна інформація:

про товар та його категорії ; про виробників товарів; про підрозділи мережі магазинів; про надходження товарів на склад; про відправку товару зі складу в підрозділ мережі; про факт продажу товару.

Розділ 2. ІНФОРМАЦІЙНЕ ТА МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ
2.1. Огляд існуючих веб-інформаційних систем

Інформаційна система Dodo IS.

Одна з компаній займається розробкою власної інформаційної системи Dodo IS. Інформаційна система Dodo IS представляє з себе веб-додаток. Доступ до системи здійснюється через звичайний веб-браузер. Сама система та база даних знаходяться на віддаленому сервері. Для роботи в системі необхідно постійне підключення комп'ютера або мобільного пристрою до Інтернету. Dodo IS призначена для управління роздрібними операціями. Бухгалтерська та фінансова звітність здійснюватимуться в спеціалізованих програмах. Між Dodo IS і спеціалізованими програмами буде налагоджено обмін даними. У Dodo IS здійснюватиметься прийом і управління замовленнями, товарний і складський облік, управління запасами, управління персоналом, клієнтська база. З інформаційною системою буде інтегрований сайт для клієнтів і мобільні додатки. Інформаційна система є одним з головних конкурентних переваг нашої бізнес-концепції. Вона замислювалася спочатку як ядро нашого бізнесу.

Спеціалізоване рішення краще універсального, оскільки будь-яке універсальне програмне рішення завжди програє додаткам, створеним під конкретний бізнес і його специфічні бізнес-процеси. До того ж на даний момент на ринку просто не існує хороших рішень для бізнесу, орієнтованого на доставку. Всі існуючі програми занадто універсальні. Або вони вимагають значно доопрацювання, або доведеться підлаштовувати бізнес під готову програмну систему. Однак мета - створити ефективний бізнес формат, який можна буде масштабувати на федеральний і навіть транснаціональний рівень. Компанія може це зробити тільки з власною системою, яка враховуватиме всю специфіку і деталі бізнесу.

Вигода в довгостроковій перспективі

Розробка власної інформаційної системи є дорогим рішенням у середньостроковій перспективі. Однак у довгостроковій перспективі з урахуванням масштабування бізнесу розробка свого ПЗ є більш вигідним

проектом, так як всі авторські права на систему належать компанії і не доведеться виробляти ліцензійні відрахування після кожного запуску нового роздрібного об'єкта.

Франчайзинг

Власна інформаційна система буде збільшувати ефективність роздрібних операцій. У майбутньому наші франчайзі отримуватимуть не тільки торгову марку, меню і стандарти, а й інформаційну систему, створену під унікальні бізнес-процеси. Це додасть франчайзингової моделі додаткову цінність. Фактично будемо ліцензувати нашу інформаційну систему разом з бізнес-системою.

Простота використання і підтримки

SaaS - технологія майбутнього. Доступність і висока швидкість Інтернету дозволить працювати з програмним забезпеченням, що не встановлюючи його на локальний комп'ютер. Сьогодні для використання інформаційної системи Dodo IS потрібно всього лише зайти через стандартний веб-бразуер на сайт і ввести логін і пароль. Необхідність адміністрування, налаштування і технічної підтримки локального мережі та програмного забезпечення відпадає.

Методика розробки інформаційної системи

Інформаційна система розробляється за принципом step by step. Проект розбивається на етапи. Завдання кожного етапу - запустити працюючий модуль. Після завершення розробки модуль відразу ж запускається в роботу. Таким чином розробники відразу ж отримують зворотній зв'язок - виправляються помилки, вносяться корективи і поліпшення. Завдання - створити максимально зручну і ефективну інформаційну систему для бізнесу, працюючи за принципом кайдзен (постійне поліпшення).

- Розроблена загальна архітектура системи і база даних.
- Розроблена адміністративна частина системи, де створюються філії, користувачі, продукти, встановлюються ціни, формується меню, налаштовується основна інформація.

- Запущений в експлуатацію перший модуль системи - "Прийом замовлення".

- Також реалізований модуль для створення і управління маркетинговими акціями.

2.2. Дерево цілей розроблюваної системи:

Рисунок 1.1 – Схема дерева цілей розроблюваної системи

2.3. Реляційна алгебра

Реляційна модель даних (РМД) - логічна модель даних, прикладна теорія побудови баз даних, яка є додатком до завдань обробки даних таких

розділів математики як теорії множин і логіка першого порядку.

На реляційної моделі даних будуються реляційні бази даних.

Реляційна модель даних включає такі компоненти:

- Структурний аспект (складова) - дані в базі даних є набором відносин.
- Аспект (складова) цілісності - відносини (таблиці) відповідають певним умовам цілісності. РМД підтримує декларативні обмеження цілісності рівня домену (типу даних), рівня відносини і рівня бази даних.
- Аспект (складова) обробки (маніпулювання) - РМД підтримує оператори маніпулювання відносинами (реляційна алгебра, реляційне числення).

Крім того, до складу реляційної моделі даних включають теорію нормалізації.

Термін "реляційний" означає, що теорія заснована на математичному понятті ставлення (relation). Як неформального синоніма терміну "відношення" часто зустрічається слово таблиця. Необхідно пам'ятати, що "таблиця" є поняття нестроге і неформальне і часто означає не "ставлення" як абстрактне поняття, а візуальне уявлення відносини на папері або екрані. Некоректне і нестроге використання терміну "таблиця" замість терміна "ставлення" нерідко призводить до нерозуміння. Найбільш часта помилка полягає в міркуваннях про те, що РМД має справу з "плоскими", або "двовимірними" таблицями, тоді як такими можуть бути тільки візуальні представлення таблиць. Відносини ж є абстракціями, і не можуть бути ні "плоскими", ні "неплоским".

Для кращого розуміння РМД слід відзначити три важливі обставини:

- модель є логічною, тобто відносини є логічними (абстрактними), а не фізичними (збереженими) структурами;
- для реляційних баз даних вірний інформаційний принцип : все інформаційне наповнення бази даних представлено одним і тільки одним способом, а саме - явним завданням значень атрибутів у кортежі відносин; зокрема, немає ніяких покажчиків (адрес), що зв'язують одне значення з іншим;

- наявність реляційної алгебри дозволяє реалізувати декларативне програмування і декларативне опис обмежень цілісності, на додаток до навігаційного (процедурним) програмування і процедурної перевірці умов.

Операції реляційної алгебри

Реляційна алгебра - замкнута система операцій над відносинами в реляційної моделі даних, або Реляці йна алгебра — відгалуження логіки першого порядку, множина відношень замкнених операторами. Оператори застосовуються до відношень, в результаті застосування отримується нове відношення.

В математиці, алгебра відношень є алгебраїчною структурою щодо математичної логіки та теорії множин. РМД стала першою працездатною моделлю даних, оскільки мала ефективний інструментарій - операції реляційної алгебри. Основною одиницею обробки є відношення, а не його кортежі.

Реляційна алгебра включає дві групи операцій.

1. Традиційні операції над множинами (модифіковані з урахуванням того, що їх операндами є відношення) - об'єднання, перетин, різниця (віднімання), декартовий твір і розподіл.

2. Спеціальні реляційні операції - вибірка, проекція, з'єднання.

2.3. Мова запитів SQL

Інформація в реляційній базі даних зберігається в таблицях і в зв’язках між таблицями. Таблиці двовимірні, мають фіксовану кількість колонок (стовп-ців, полів) та довільну кількість рядків (записів). Кожна колонка має найменування і містить дані певного типу. Для ідентифікації запису застосовується первинний ключ – одне або декілька полів, які містять унікальні значення в межах таблиці. В архітектурі клієнт-сервер, клієнт посилає на сервер запит і отримує від нього відповідь – результат виконання запиту. Найбільшого поширення набула мова запитів SQL (акронім від англійського Structured Query Language – структурована мова запитів). Приклад простої команди на мові SQL, яка показує всі колонки та всі рядки з таблиці Student: *SELECT * FROM Student;*

У свою чергу, мова SQL поділяється на три підмножини: DDL (Data Definition Language), DML (Data Manipulation Language) і DCL (Data Control Language). DDL визначає набір команд, за допомогою яких у базі даних створюються структурні об'єкти, т.зв. метадані – таблиці, домени, зовнішні ключі, індекси, збережені процедури і т.п. Мовою DML пишуться запити на отримання даних з бази, а також на вставку, зміну або видалення записів у таблиці. Мова DCL містить набір операторів для розмежування доступу до даних для різних користувачів СКБД.

Типи даних БД

При створенні таблиці потрібно задати тип даних для кожної колонки. До наших послуг набір вбудованих типів: INTEGER і SMALLINT

Цілочисельні числа, 4-х і 2-х байтові відповідно, дозволяють зберігати значення в діапазоні: INTEGER: -2147483648 .. 2147483647, SMALLINT: -32 768 .. 32 767.DOUBLE PRECISION і SINGLE PRECISION.

Числа з плаваючою крапкою мають довжину 8 і 4-и байта відповідно. Діапазони значень: DOUBLE PRECISION: 5.0×10^{-324} .. 1.7×10^{308} (15-16 значущих цифр),SINGLE PRECISION: 1.5×10^{-45} .. 3.4×10^{38} (7-8 значущих цифр).NUMERIC і DECIMAL

Числа з фіксованою крапкою. При визначенні типу в дужках вказується загальна кількість збережених цифр у числі і кількість знаків після десяткової крапки. Наприклад, поле визначене як NUMERIC (6, 2) дозволяє зберігати числа від -9999.99 до 9999.99. Тип даних NUMERIC (18, 0) визначає 64-х бітове ціле число. DATE, TIME, TIMESTAMP. Визначає дату, час і дату з часом відповідно. CHAR, VARCHAR

Визначає рядки. При створенні колонки в дужках вказується максимальна довжина рядка, наприклад: VARCHAR (200). Допустимі значення довжини – від 1 до 32000 символів. Тип даних CHAR завжди зберігає рядки фіксованої довжини. Рядки меншої довжини доповнюються справа пробілами. Тип VARCHAR зберігає стільки символів, скільки їх було поміщено в базу даних командою INSERT або UPDATE. При створенні рядкового поля можна вказати

кодову таблицю і порядок звірення. Наприклад: CHARACTER SET WIN1251 COLLATE PXW_CYRL. Від вказаної кодової таблиці залежить скільки байт буде виділятися на зберігання одного символу.

BLOB. Визначає двійковий об'єкт. Застосовується для зберігання в базі даних малюнків, файлів, великих текстів, потоків даних і т.п.

Основні оператори мови SQL

SELECT – оператор мови SQL, котрий повертає рядки з однієї чи багатьох таблиць. Повний синтаксис оператора SELECT є складним, проте його можна описати так:

SELECT список_вибірки
[INTO нова_табличка]
FROM таблиця
[WHERE умови_пошуку]
[GROUP BY групувати_по_умові]
[HAVING умови_пошуку]
[ORDER BY сортувати_по_умові [ASC | DESC]]

Повертає нуль або більше рядків з однієї або більше таблиць, тимчасових таблиць, або ж представлень бази даних. У більшості застосунків, SELECT – найчастіша команда Data Manipulation Language (DML). Оскільки, SQL не є процедурною мовою, запит SELECT описує кінцеві дані, однак, не вказує, які саме операції слід виконати для отримання цих даних: завдання покладається на систему керування базами даних, яка здатна самостійно оптимізовувати необхідні для отримання результату операції. Запит SELECT має такі допоміжні параметри: WHERE вказує, які рядки слід вибрати; GROUP BY гуртує рядки, що мають спільну властивість таким чином, щоб функція агрегації могла бути застосована до кожної групи; HAVING вибирає з груп означених оператором GROUP BY; ORDER BY вказує порядок повернення рядків.

Розділ 3. ПРОГРАМНЕ ТА ТЕХНІЧНЕ ЗАБЕЗПЕЧЕННЯ

3.1. Опис PHP Designer

PHP Designer являє собою зручний засіб, призначене для написання, редагування, виправлення, аналізу та компіляції сторінок і веб-додатків, написаних на мові PHP. Дана програма відмінно підійде як початківцям кодерам, так і просунутим розробникам. Ключовою особливістю PHP Designer є те, що в програмі реалізована підтримка не тільки PHP, але і MySQL, HTML, CSS, C, Python JavaScript, Ruby і так далі. Додаток має вбудоване засіб для підсвічування синтаксису, а також включає в себе бібліотеки, що містять кілька тисяч функцій, які дуже знадобляться вам в процесі написання програмного коду. В цілому, досить зручна і функціональна середовище розробки з великою кількістю підтримуваних мов програмування.

Ключові особливості та функції програми:

- підсвічування синтаксису для всіх підтримуваних програмою мов;
- функція автоматичного завершення коду для мов PHP, JavaScript і CSS;
- підтримка величезної кількості бібліотек;
- вбудована система допомоги;
- вбудований інспектор HTML;
- можливість налагодження й профілювання скриптів з використанням Xdebug;
- вбудований FTP-клієнт;
- зручна навігація по файлах проекту.

3.2. Мова програмування PHP

PHP (PHP: Hypertext Preprocessor — PHP: гіпертекстовий препроцесор) – попередня назва: Personal Home Page Tools — скриптова мова програмування, була створена для генерації HTML-сторінок на стороні веб-сервера. PHP є однією з найпоширеніших мов, що використовуються у сфері веб-розробок (разом із Java, .NET, Perl, Python, Ruby). PHP підтримується переважною

більшістю хостинг-провайдерів. PHP — проект відкритого програмного забезпечення.

PHP інтерпретується веб-сервером в HTML-код, який передається на сторону клієнта. На відміну від скриптової мови JavaScript, користувач не бачить PHP-коду, бо браузер отримує готовий html-код. Це є перевага з точки зору безпеки, але погіршує інтерактивність сторінок. Але ніщо не забороняє використовувати PHP для генерування і JavaScript-кодів які виконуються вже на стороні клієнта.

Через стандарт відкритого інтерфейсу зв'язку з базами даних (Open Database Connectivity Standard — ODBC) можна підключатися до всіх баз даних, до яких існує драйвер.

Традиційність

Мова PHP здаватиметься знайомою програмістам, що працюють в різних областях. Багато конструкцій мови запозичені з C, Perl. Код PHP дуже схожий на той, який зустрічається в типових програмах на C або Pascal. Це помітно знижує початкові зусилля при вивченні PHP. PHP — мова, що поєднує переваги Perl і C і спеціально спрямована на роботу в Інтернеті, мова з універсальним і зрозумілим синтаксисом. І хоча PHP є досить молодою мовою, вона здобула таку популярність серед web-програмістів, що в наш час є мало не найпопулярнішою мовою для створення веб-застосунків (скриптів).

Наявність вихідного коду та безкоштовність.

Стратегія Open Source, і розповсюдження початкових текстів програм в масах, безсумнівно справили благотворний вплив на багато проектів, в першу чергу — Linuxхоч і успіх проекту Apache сильно підкріпив позиції прихильників Open Source. Сказане відноситься і до історії створення PHP, оскільки підтримка користувачів зі всього світу виявилася дуже важливим чинником в розвитку проекту PHP.

Ефективність

Ефективність є дуже важливим чинником у програмуванні для середовищ розрахованих на багато користувачів, до яких належить і web. Важливою

перевагою PHP є те, що ця мова належить до інтерпретованих. Це дозволяє обробляти сценарії з достатньо високою швидкістю. За деякими оцінками, більшість PHP-сценаріїв (особливо не дуже великих розмірів) обробляються швидше за аналогічні їм програми, написані на Perl. Проте хоч би що робили розробники PHP, виконувані файли, отримані за допомогою компіляції, працюватимуть значно швидше — в десятки, а іноді і в сотні разів. Але продуктивність PHP достатня для створення цілком серйозних веб-застосунків.

Синтаксис

Всі сценарії оформляються у вигляді блоків коду. Ці блоки можуть бути поміщені в HTML-код, але відділені від нього відповідними обмежувачами. Код PHP в HTML повинен знаходитись між початковим тегом <?php та кінцевим ?> (або між <script language="php"> та </script>) Бажаним варіантом виділення PHP коду є варіант <?php ?>, оскільки саме такі початковий та кінцевий теги дозволять використовувати PHP код в документах, які відповідають правилам XML.[18,]

Також можна користуватися скороченим записом: <? ?> (інколи потрібно активізувати даний стиль внісши вручну зміни в файл php.ini: змінна *short_open_tag* повинна мати значення *On*) і записом в стилі ASP: <% %> (в php.ini змінна *asp_tags* повинна мати значення *On*). Проте стиль ASP не рекомендується і очікується, що він буде відсутній у PHP6.

PHP виконує код, що знаходиться в середині обмежувачів, таких як `<?php ?>`. Все, що знаходиться поза межами обмежувачів виводиться без змін. Таким чином виконується вставка PHP коду в HTML код. Наприклад, код html-сторінки з попереднім прикладом виглядатиме так:

```
<html>
 <head>
  <title>Тестуємо PHP</title>
 </head>
 <body>
  <?php echo 'Hello, world!'; ?>
 </body>
</html>
```

Робота з рядками

Рядки ділять на два класи — рядки, що підлягають аналізу, та ті, що не підлягають. Перший клас досліджується інтерпретатором на наявність посилань на інші змінні, і за умови їхньої наявності робиться підстановка значень у відповідне місце. Крім того, клас дозволяє проводити маніпуляції з керівними символами. Символ рядка може мати лише одне з 256 значень, але є можливість працювати з багатобайтовими символами. Доступ до символів рядка можливий з використанням синтаксису, схожого на доступ до елементів масивів.

PHP надає широкий спектр функцій для пошуку та заміни тексту в рядках. Для цього використовують як традиційний підхід, так і спеціальний, що базується на використанні регулярних виразів. При цьому в мові реалізована підтримка двох видів регулярних виразів — Perl-сумісні та POSIX-сумісні, що розрізняються за синтаксисом та особливостями роботи.

Конструкції мови

Оператори в сенсі мови дозволяють виконувати відповідну дію над одним чи кількома операндами. Оператори бувають трьох типів: унарні, бінарні та тернарні. Оператори, як і в інших мовах характеризуються не лише дією, а й асоціативністю та пріоритетністю. Особливістю булевих операцій порівняння розрізнення двох класів з врахуванням типу і без врахування типу, при якому відбувається приведення до відповідного типу. Округлення відбуваються завжди в меншу сторону. В мові реалізовані особливі класи операторів —виконання, управління помилками та перевірки приналежності до класу.

Функції в сенсі мови є контейнерами коду, причому можливе включення інших функцій та класів. На цьому і базується можливість умовного визначення функції. В цьому випадку висувається вимога попередньої декларації викликаної функції, що не обов'язкове в інших випадках. Можливості перевизначення чи деактивації функції не існує. Результат, який повертає функція може мати будь-який тип.

В мові реалізована функціональність посилань. Можливо створити скільки завгодно псевдонімів, що посилаються на єдиний сегмент даних. При вивільненні будь-якого з псевдонімів, сегмент даних залишається в пам'яті до моменту завершення сценарію або вивільнення усіх посилань.

Що стосується функцій в PHP, то замість прийнятого в багатьох мовах принципу перевантаження функцій, що дозволяє змінити хід виконання певної функції в залежності від типу та кількості переданих параметрів, використовується метод динамічних аргументів. Це дає змогу не визначати кількість параметрів для функцій при їх оголошенні, а працювати із тими аргументами, які були отримані на момент виклику функції. У тілі функції можливо отримати кількість переданих їй аргументів і проводити відповідні маніпуляції. При оголошенні функції звичайним чином, можливе задання значень аргументів за замовчуванням. Функції можуть повертати лише одне значення, проте це обмеження можна оминути, використавши не лише масиви, а й посилання. Передача аргументів за посиланням неможлива під час виконання та оголошення функції.

Після виконання сценаріїв, простір пам'яті, займаної ними очищується збирачем сміття. Проте, за необхідності можливе виконання очищення пам'яті від надлишкових сегментів даних під час виконання скриптів. Використання функцій очищення пам'яті є невиправданим, хоча така можливість існує.

Для побудови програмних комплексів можна використовувати модульний підхід, виконуючи розділення різнорідного коду. При потребі, можливе виконання під'єднання необхідних модулів, причому операція виконання може бути і умовною. Під'єднані до скрипта файли можуть повертати значення.

3.3. Jquery

jQuery — популярна JavaScript-бібліотека з відкритим серцевим кодом. Вона була представлена у січні 2006 року у BarCamp NYC Джоном Ресіґом (John Resig). Згідно з дослідженнями організації W3Techs, JQuery використовується понад половиною від мільйона найвідвідуваніших сайтів. jQuery є

найпопулярнішою бібліотекою JavaScript, яка посилено застосовується на сьогоднішній день.

jQuery є вільним програмним забезпеченням під ліцензією MIT (до вересня 2012 було подвійне ліцензування під MIT та GNU General Public License другої версії).

Синтаксис jQuery розроблений, щоб зробити орієнтування у навігації зручнішим завдяки вибору елементів DOM, створенню анімації, обробки подій, і розробки AJAX-застосунків. jQuery також надає можливості для розробників, для створення плагінів у верхній частині бібліотеки JavaScript. Використовуючи ці об'єкти, розробники можуть створювати абстракції для низькорівневої взаємодії та створювати анімацію для ефектів високого рівня. Це сприяє створенню потужних і динамічних веб-сторінок.

Основне завдання jQuery — це надавати розробнику легкий та гнучкий інструментарій кросбраузерної адресації DOM об'єктів за допомогою CSS та XPath селекторів. Також даний фреймворк надає інтерфейси для Ajax-застосунків, обробників подій і простої анімації.

Принцип роботи jQuery полягає в використанні класу (функції), який при звертанні до нього повертає сам себе. Таким чином, це дозволяє будувати послідовний ланцюг методів.

Бібліотека jQuery є JavaScript файлом, яка включає всю його DOM, події(events), ефекти(effects), і Ajax функції. Вона може бути додана до web-сторінки посиланням на локальну копію, або на одну з копій доступних на публічному сервері (наприклад Google або Microsoft CDN).

```
<script type="text/javascript" src="jquery.js"></script>
$('#test') //знаходимо елемент з id="test"
    .text('Клікни по мені')  //встановлюємо текст елемента рівним "Клікни по мені"
    .addClass('myAlert')//додаємо клас "myAlert"
    .css('color','red')//встановлюємо колір тексту червоним
    .attr('alert','Привіт, світе!') // додаємо атрибут "alert" із значенням "Привіт, світе!"
```

```
    .bind(// додаємо в обробник події click функцію, яка відкриє
модальне 'click',// вікно із текстом, що вказаний в атрибуті "alert"
("Привіт, світе!")
        function(){alert($(this).attr('alert'))}
    );
```

3.4. Ajax

AJAX - (Asynchronous JavaScript And XML) - підхід до побудови інтерфейсів веб-застосунків, за яких веб-сторінка, не перезавантажуючись, у фоновому режимі надсилає запити на сервер і сама звідти довантажує потрібні користувачу дані. AJAX — один з компонентів концепції DHTML.

Про AJAX заговорили після появи в лютому 2005-го року статті Джесі Джеймса Гарретта (Jesse James Garrett) «Новий підхід до веб-застосунків». AJAX — не самостійна технологія. Це ідея.

AJAX — це не самостійна технологія, а швидше концепція використання декількох суміжних технологій. AJAX підхід до розробки призначених для користувача інтерфейсів комбінує кілька основних методів і прийомів:

Використання DHTML для динамічної зміни змісту сторінки.

Використання XMLHttpRequest для звернення до сервера «на льоту», не перезавантажуючи всю сторінку повністю

Альтернативний метод динамічне підвантаження коду JavaScript в тег <SCRIPT> з використанням DOM, що здійснюється із використанням формату JSON).

Динамічне створення дочірніх фреймів.

Використання цих підходів дозволяє створювати набагато зручніші веб-інтерфейси користувача на тих сторінках сайтів, де необхідна активна взаємодія з користувачем. AJAX — асинхронний, тому користувач може переглядати далі контент сайту, поки сервер все ще обробляє запит. Браузер не перезавантажує web-сторінку і дані посилаються на сервер без візуального підтвердження (крім випадків, коли ми самі захочемо показати процес з'єднання з сервером). Використання AJAX стало найпопулярніше після того, як компанія Google почала активно використовувати його при створенні своїх

сайтів, таких як Gmail, Google Maps і Google Suggest. Створення цих сайтів підтвердило ефективність використання даного підходу.

3.5. HTML

HTML (HyperTextMarkupLanguage — мова розмітки гіпертекстових документів) — стандартна мова розмітки веб-сторінок в Інтернеті. Більшість веб-сторінок створюються за допомогою мови HTML (або XHTML). Документ HTML оброблюється браузером та відтворюється на екрані у звичному для людини вигляді.

HTML є похідною мовою від SGML, успадкувавши від неї визначення типу документу та ідеологію структурної розмітки тексту.

HTML впроваджує засоби для:

- створення структурованого документу шляхом позначення структурного складу тексту: заголовки, абзаци, списки, таблиці, цитати та інше;
- отримання інформації із Всесвітньої мережі через гіперпосилання;
- створення інтерактивних форм;
- включення зображень, звуку, відео, та інших об'єктів до тексту.

Для поліпшення взаємодії, SGML вимагає аби кожна похідна мова (HTML у тому числі) визначала свою кодову таблицю для кожного документа, яка складається зрепертуара (перелік різноманітних символів) та позиції символа (перелік цифрових посилань на символи з репертуара). Кожен документ HTML — це послідовність символів з репертуара.

HTML використовує найповнішу кодову таблицю UCS (англ. Universal Character Set — Універсальний Набір Символів).

Проте, однієї кодової таблиці недостатньо для того, щоб браузери могли правильно відтворювати документи HTML. Для цього браузерам потрібно «знати» специфічну кодову таблицю документа, яку автор має зазначати завжди в елементі `meta` із параметром `charset`. За замовчуванням вибрана кодова таблиця ISO-8859-1, відома також як Latin-1.

AppServ

AppServ - це повноцінний набір програм, які необхідні для запуску http сервера на операційній системі Windows. До складу пакету входить такі додатки як Apache, PHP, MySQL, phpMyAdmin та інші програми. AppServ дозволяє запустити власне сам web сервер. MySQL - це сервер баз даних в якому зберігатиметься інформація баз даних. PHP - це мова програмування вебпріложеній. phpMyAdmin - це панель управління сервером баз даних.

AppServ автоматично встановлює і налаштовує ці програми в одній папці. При цьому кожну окрему програму з набору можна встановити як системну службу. Більшість тонких налаштувань сесрвера потрібно проводити за допомогою редагування конфігураційних файлів. Отримати доступ до сервера можна набравши в адресному рядку браузера 127.0.0.1 або localhost.

СКБД MySQL

MySQL – вільна система керування реляційними базами даних.

Ця система керування базами даних (СКБД) з відкритим кодом була створена як альтернатива комерційним системам. MySQL з самого початку була дуже схожою на mSQL, проте з часом вона все розширювалася і зараз MySQL– одна з найпоширеніших систем керування базами даних. Вона використовується, в першу чергу, для створення динамічних веб-сторінок, оскільки має чудову підтримку з боку різноманітних мов програмування.

MySQL– компактний багатонитковий сервер баз даних. Характеризується великою швидкодією, стійкістю і простотою використання.

MySQL був розроблений компанією «TcX» для підвищення швидкодії обробки великих баз даних.

MySQL вважається гарним рішенням для малих і середніх застосувань. Вихідні коди сервера компілюються на багатьох платформах. Найповніше можливості сервера виявляються в UNIX-системах, де є підтримка багатопотоковості, що підвищує продуктивність системи в цілому.

Для некомерційного використання MySQL є безкоштовним. Можливості сервера MySQL:

- простота у встановленні та використанні;
- підтримується необмежена кількість користувачів, що одночасно працюють із БД;
- кількість рядків у таблицях може досягати 50 млн.;
- висока швидкість виконання команд;
- наявність простої і ефективної системи безпеки.

MySQL виникла як спроба застосувати mSQL до власних розробок компанії: таблиць, для яких використовувалися ISAM– підпрограми низького рівня. У результаті був вироблений новий SQL-інтерфейс, але API-інтерфейс залишився в спадок від mSQL. В січні-лютому 2008 SunMicrosystems придбала розробника системи керування базами данних MySQL за $1 млрд. Після поглинання у 2009 році SunMicrosystemskомпанієюOracleCorporationMySQL стала власністю Oracle.

Ліцензування MySQL

MySQL має подвійне ліцензування. MySQL може розповсюджуватися відповідно до умов ліцензії GPL. Але за умовами GPL, якщо якась програма використовує бібліотеки MySQL, то вона теж повинна розповсюджуватися за ліцензією GPL. Проте це може розходитися з планами розробників, які не бажають відкривати вихідні тексти своїх програм. Для таких випадків передбачена комерційна ліцензія компанії MySQL AB, яка також забезпечує якісну сервісну підтримку. В разі використання та розповсюдження програмного забезпечення з іншими вільними ліцензіями, такими як BSD, Apache, MIT та інші, MySQL дозволяє використання бібліотек MySQL за ліцензією GPL.

Версія MySQL 5.5 вийшла 27грудня 2012 року. У цій версії була значно розширена функціональність, що ставить MySQL в один ряд із комерційними СКБД. Якщо попередні версії СКБД MySQL звинувачували у недостатній підтримці стандарту SQL, то із появою п'ятої версії цієї популярної системи керування базами даних появилась практично повна підтримка стандарту SQL. MySQL 5.5 містить такі нововведення: збережувані процедури та функції;

обробники помилок; курсори; тригери; представлення; інформаційна схема (так званий системний словник, що містить метадані).

3.6. Розробка Web-сайту

Керуючись даними принципами розробки інтерфейсу, було вирішено зробити ставку на простоту та інформативність, що б користувач, потрапляючи на сайт, міг отримувати чітку інформацію про товар, його наявність, магазини і постачальників, також щоб було легко редагувати дані.

Враховуючи описані відомості про Інтернет-системи було розроблено веб-інформаційну систему аналізу роботи магазинів спортивних товарів з використанням мови PHP та розроблена функціонуюча база даних товарів мережі магазинів з використанням MySQL.

База даних deep_dp містить 7 основних таблиць [18] :

1. dp_categories - категорії товарі
2. dp_link - мережа
3. dp_store – підрозділи
4. dp_sales - продажі
5. dp_ stock - склад
6. dp_goods - товари
7. dp_ vendor – виробники

Таблиця dp_categories має 2 поля:

Поле	Тип
id	int(255)
name	varchar(255)

Таблиця 1. (dp_categories)

- id;
- name.

Поле id – ключове. Поле name містить назву категорії. Дані цієї таблиці використовуються для створення випадаючого списку, який містить всіх виробників і дозволяє фільтрувати список товарів за виробником.

Таблиця dp_ vendor має 6 полів:

Поле	Тип
id	int(255)
name	varchar(255)
subvendor	varchar(255)
adress	varchar(255)
email	varchar(255)
telphone	varchar(255)

Таблиця 2. (dp_ vendor)

- id – ключове;
- name – містить назву виробника;
- subvendor – фірма-постачальник, якщо поставки товарів здійснюються через посередника.
- adress – адреса:
- email – електронна пошта;
- telphone – контактний телефон

Таблиця dp_goods призначена для інформації про товари, якими торгує мережа магазинів. Містить 6 полів:

Поле	Тип
id	int(255)
code	varchar(255)
name	varchar(255)
id_category	int(255)
id_vendor	int(255)
price	varchar(255)

Таблиця 3. (dp_goods)

- id – ключове поле;
- code – код товару. Код товару, який присвоюється товару всередині мережі магазинів.
- name – назва товару;
- id_category – поле зовнішнього ключа, через яке реалізується зв’язок з таблицею dp_categories. Дозволяє фільтрувати товари за категоріями.
- id_vendor – поле зовнішнього ключа, через яке реалізується зв’язок з таблицею dp_ vendor. Дозволяє фільтрувати товари за виробниками.
- price – ціна товару. Значення присвоюється підчас приймання товару на склад.

Таблиця dp_ stock призначена для інформації про товари, які знаходяться на складі мережі спортивних магазинів. Містить 3 поля:

Поле	Тип
id	int(255)
id_good	int(255)
count	int(255)

Таблиця 4. (dp_ stock)

- id – ключове поле;
- id_good – містить код товару, який знаходиться на складі. Є полем зовнішнього ключа, через яке реалізується зв’язок з таблицею dp_goods.
- count – кількість товарів даного наіменування на складі.

Таблиця dp_link призначена для інформації про товари, які знаходяться в мережі – тобто у будь-якому з підрозділів -- спортивних магазинів. Містить 4 поля:

Поле	Тип
id	int(255)
id_good	int(255)
id_store	int(255)
count	int(255)

Таблиця 5. (dp_link)

- id – ключове поле;
- id_good – містить код товару, який знаходиться на складі. Є полем зовнішнього ключа, через яке реалізується зв’язок з таблицею dp_goods;
- id_store – код підрозділу. Несе інформацію про те, у якому з підрозділів знаходиться товар;
- count – кількість товарів даного наіменування у даному підрозділі.

Таблиця dp_sales містить інформацію про продажі у кожному з магазинів мережі. Має 5 полів:

Поле	Тип
id	int(255)
id_good	int(255)
id_store	int(255)
count	int(255)
datetime	datetime

Таблиця 6. (dp_sales)

- id – ключове поле;
- id_good – код проданого товару. Є полем зовнішнього ключа, через яке реалізується зв’язок з таблицею dp_goods;
- id_store – код підрозділу, у якому був проданий товар;
- count – кількість проданих одиниць даного товару;
- datetime – час і дата продажу.

Таблиця dp_store містить інформацію про підрозділи мережі магазинів. Має 4 поля:

Поле	Тип
id	int(255)
name	varchar(255)
telphone	varchar(255)
adress	varchar(255)

Таблиця 7. (dp_store)

- id – ключове поле;
- name – містить назву підрозділу;
- telphone – контактний телефон;
- adress – адреса;

Також 4 системні таблиці для керування інформаційною системою, адміністрування і сортування даних.

Таблиця 8. Каталог системних таблиць

3.7. Принцип роботи Інтернет-магазину

В інформаційній системі ми маємо змогу редагувати дані в особистому кабінеті, підключати нові модулі, чи редагувати вже існуючі, надавати права доступу користувачам, переглядати, додавати, редагувати, видаляти дані про існуючі товари, про товари на складі, наявні в магазинах, про виробників та підрозділи мережі, слідкувати за продажами.

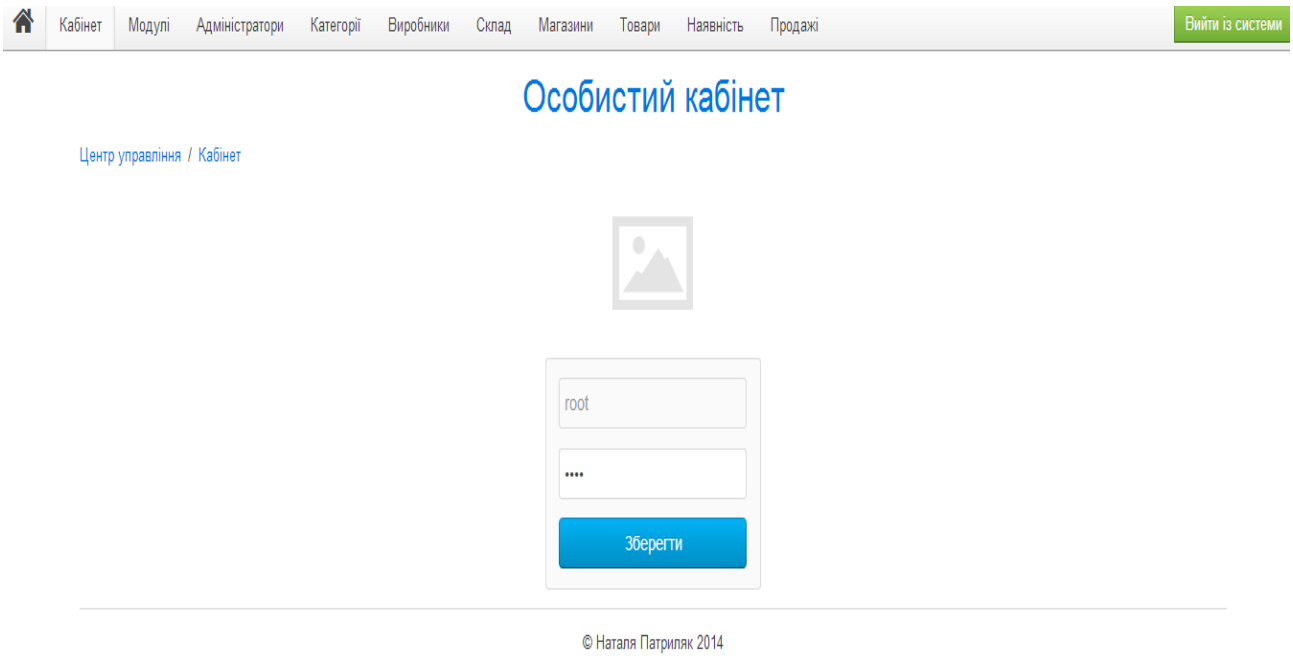

Рисунок 3.1 – Вигляд інформаційної системи

Інформація про товари зберігається в таблиці “Товари” (рис.3.2), з якої ми можемо дізнатися про назву товару, опис, модель, виробника, код товару, наявність товару на складі. Ключове поле – “Код товару”, числового типу.

В таблиці “Товари” , а також в наступній таблиці “Редагування даних з таблиці «Товари»” (рис. 3.3) присутні додаткові поля. Оскільки вони використовуються лише для функціонування CMS Joomla, то опису не потребують.

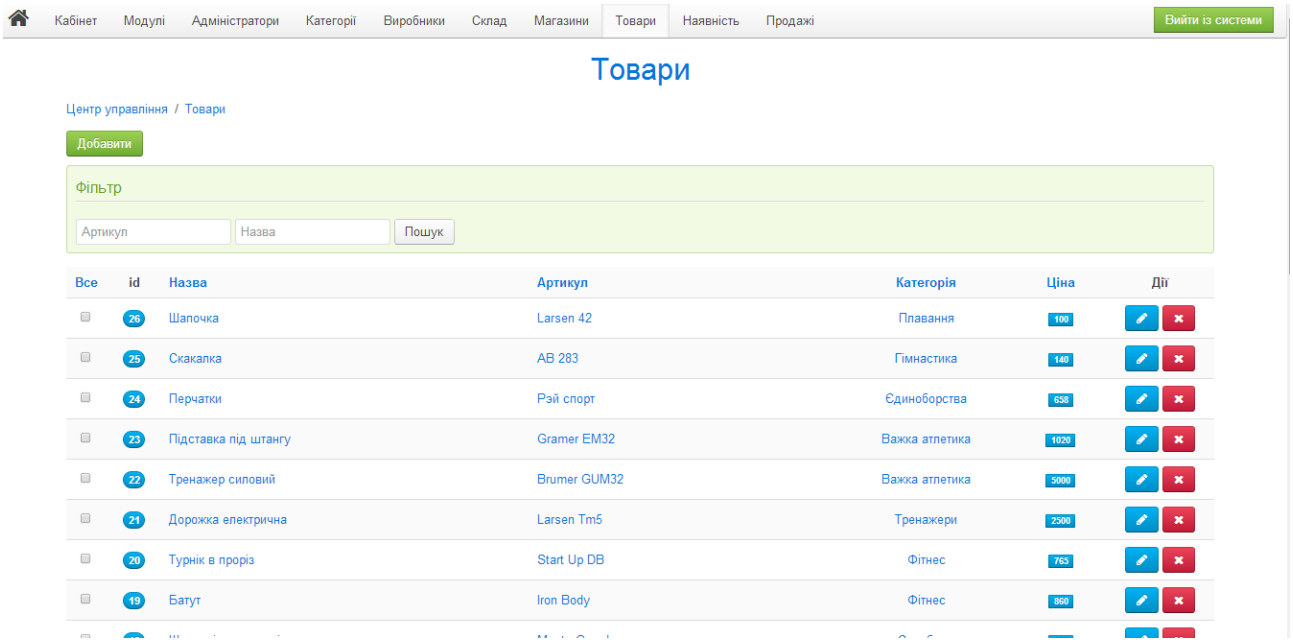

Рисунок 3.2 – Перегляд даних з таблиці «Товари»

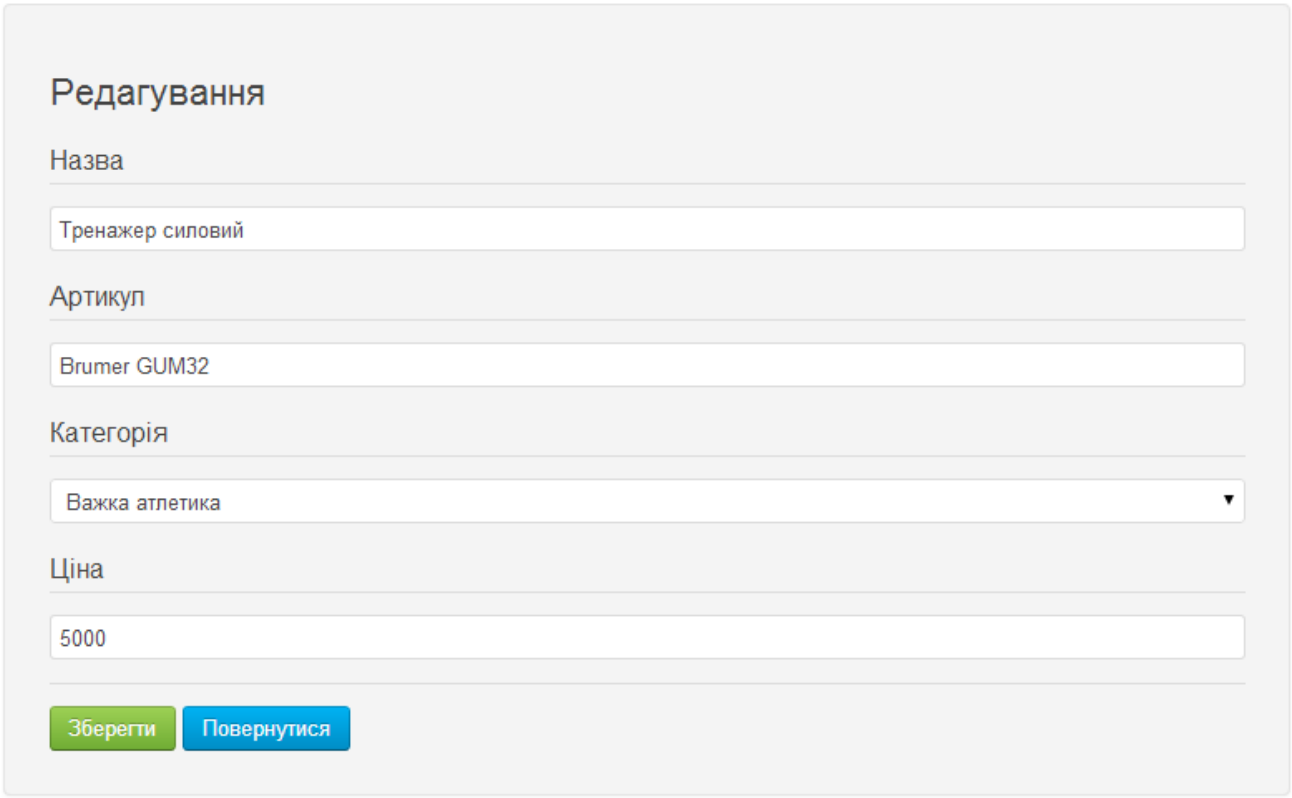

Рисунок 3.3 – Редагування даних з таблиці «Товари»

Інформація про виробників та їх продукти розміщена в таблиці “Виробники” (рис. 3.4). Таблиця містить назву виробника, а також контактні дані. Ключове поле – “Код виробника”, числового типу.

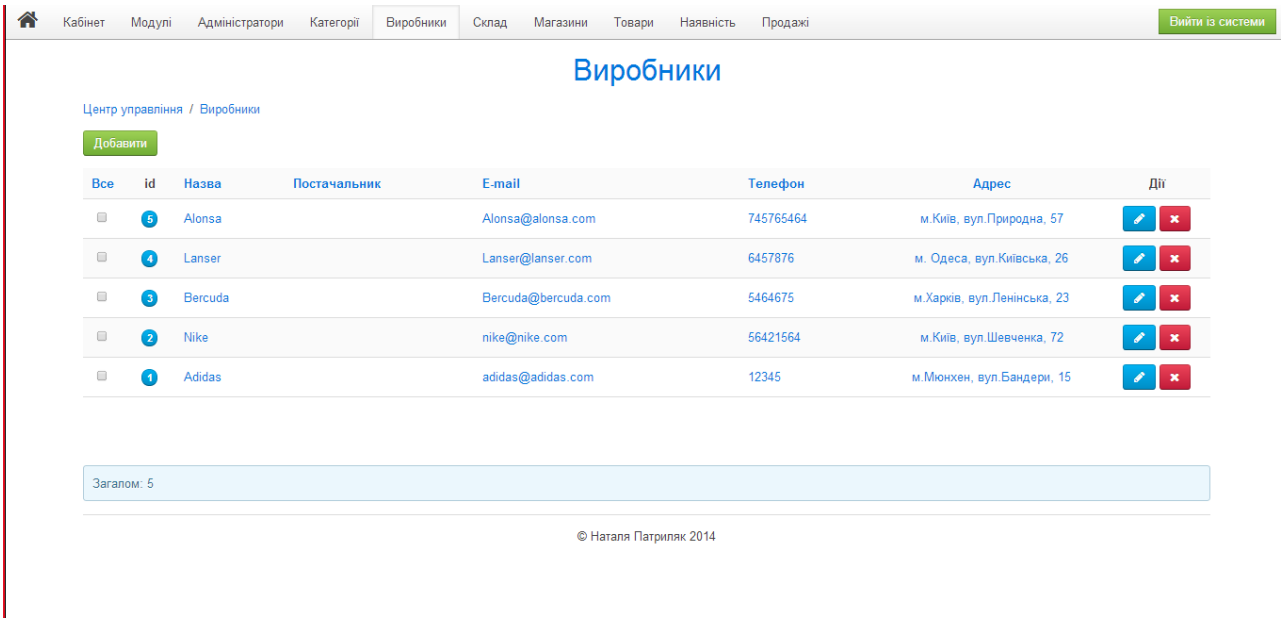

Рисунок 3.4 – Перегляд даних про виробників

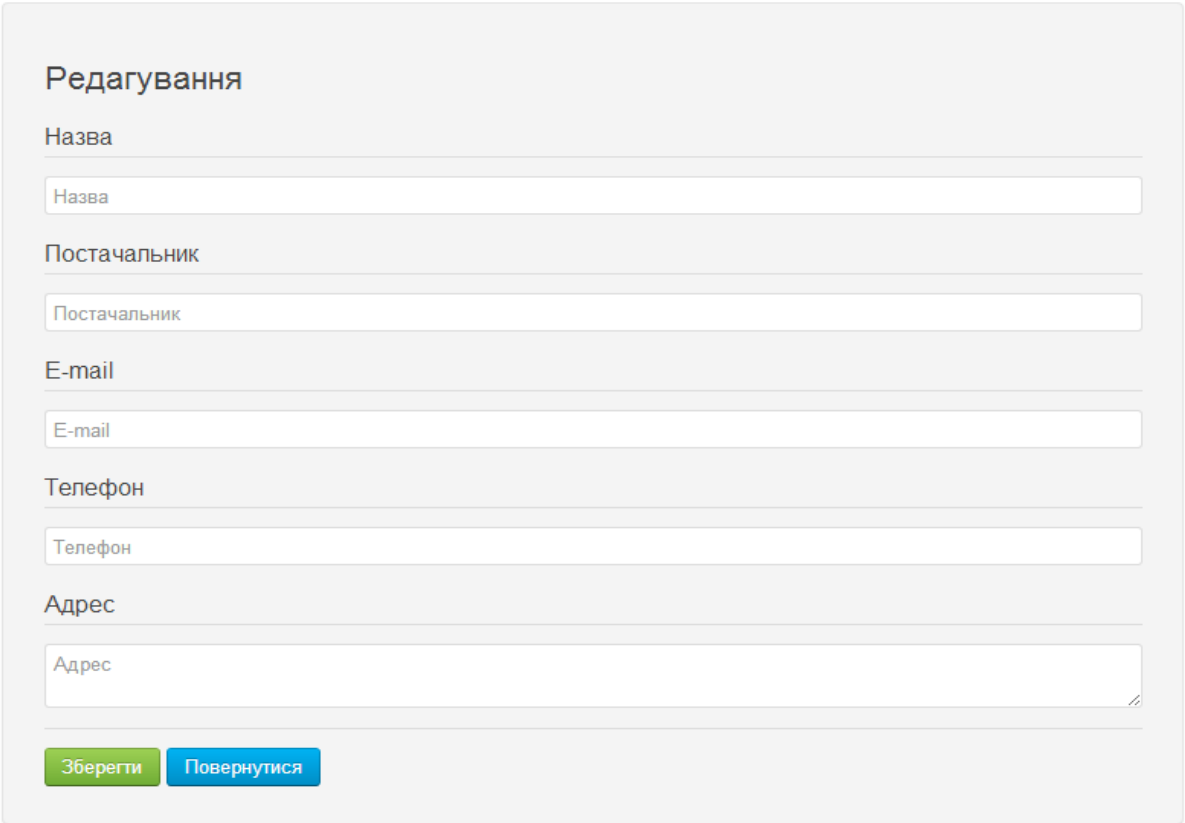

Рисунок 3.5 – Додавання нового запису до таблиці «Виробники»

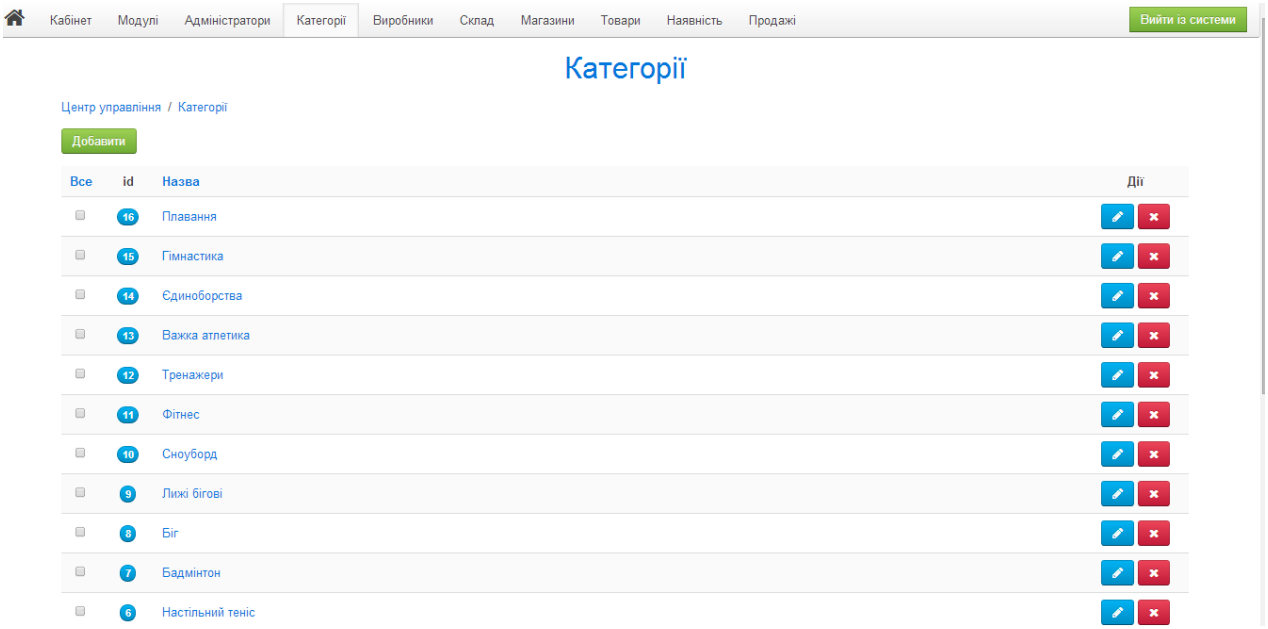

Рисунок 3.6 – Перегляд таблиці з даними про категорії товарів

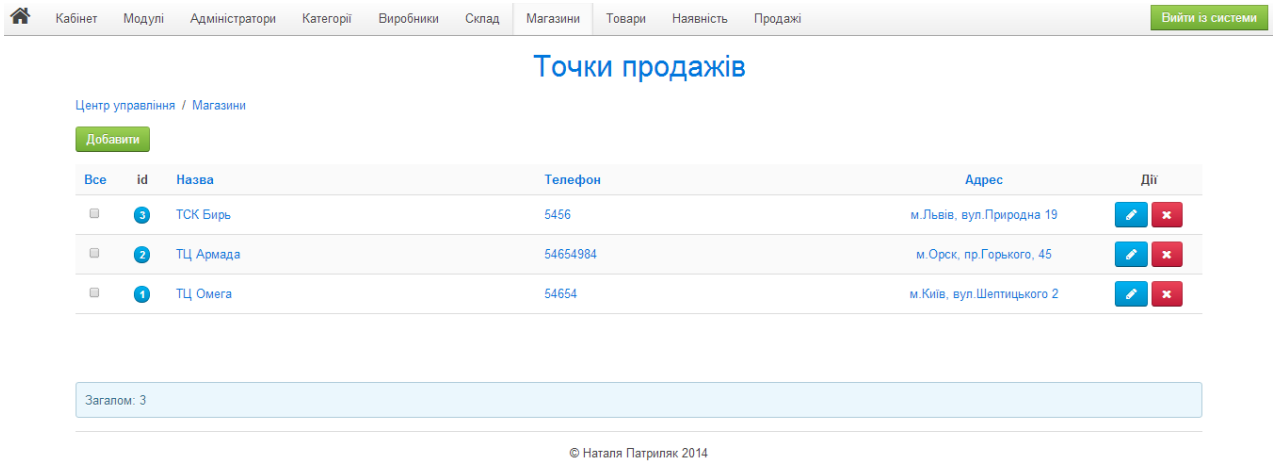

Рисунок 3.7 – Перегляд таблиці з даними про підрозділи мережі

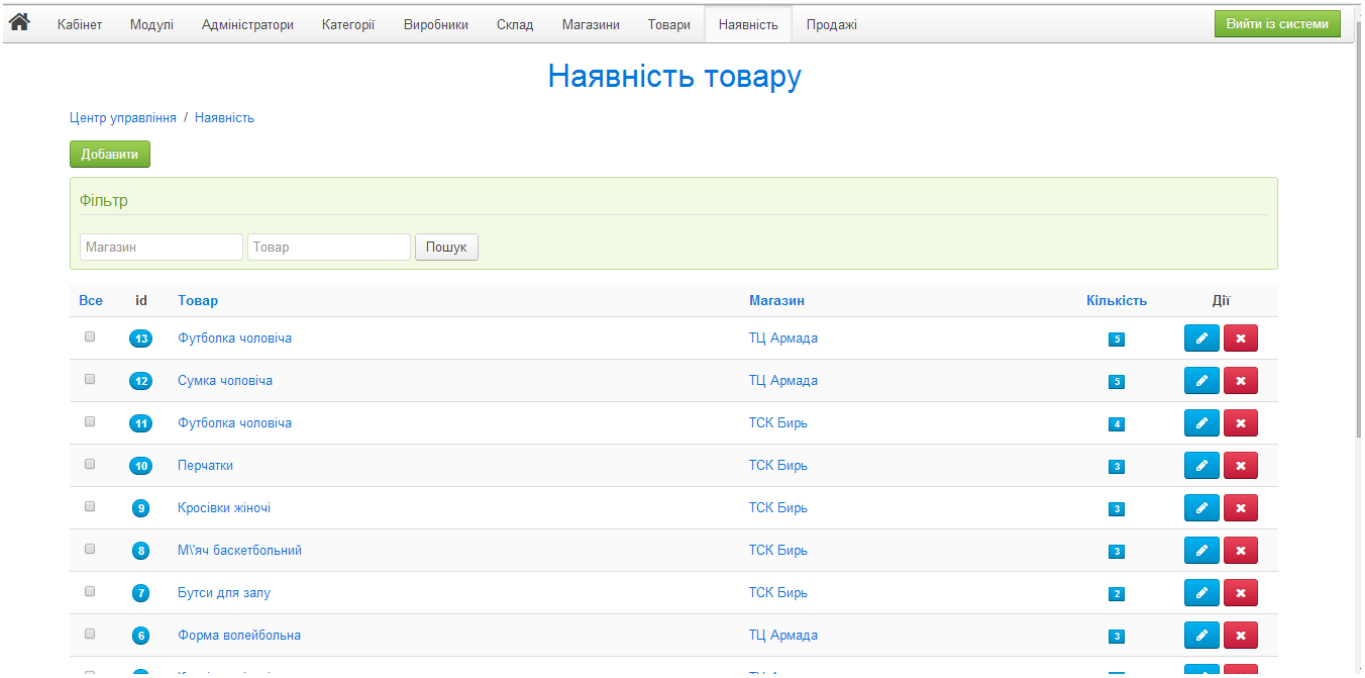

Рисунок 3.8–- Наявність товару

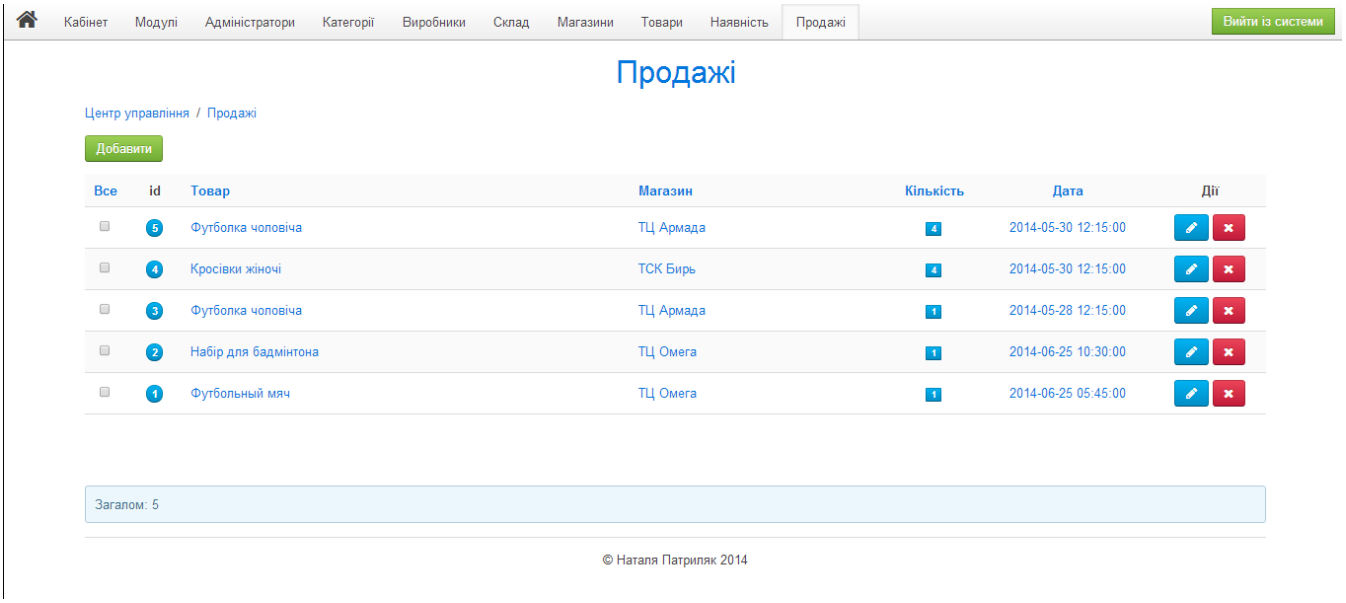

Рисунок 3.9 – Продажа товарів

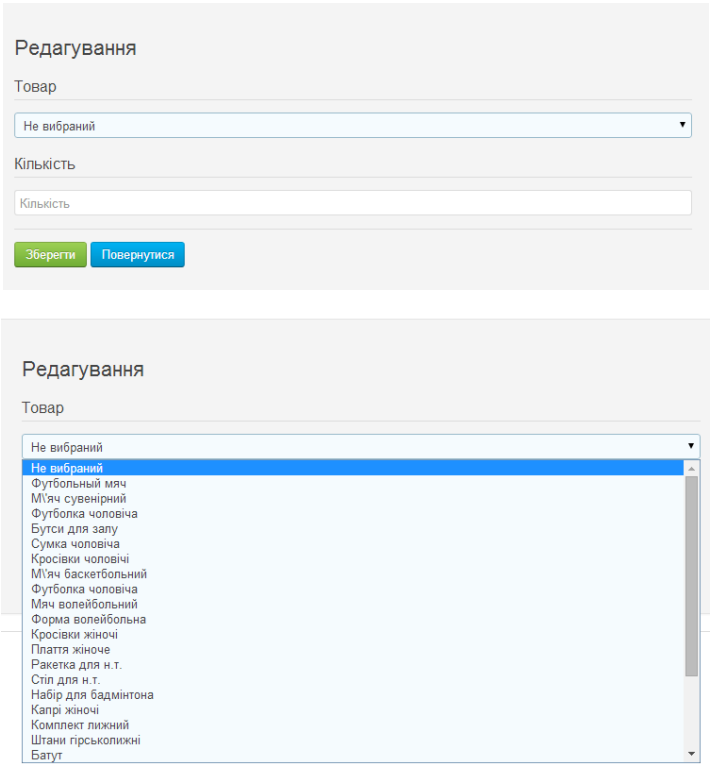

Рисунок 3.10 – Подання товарів на склад

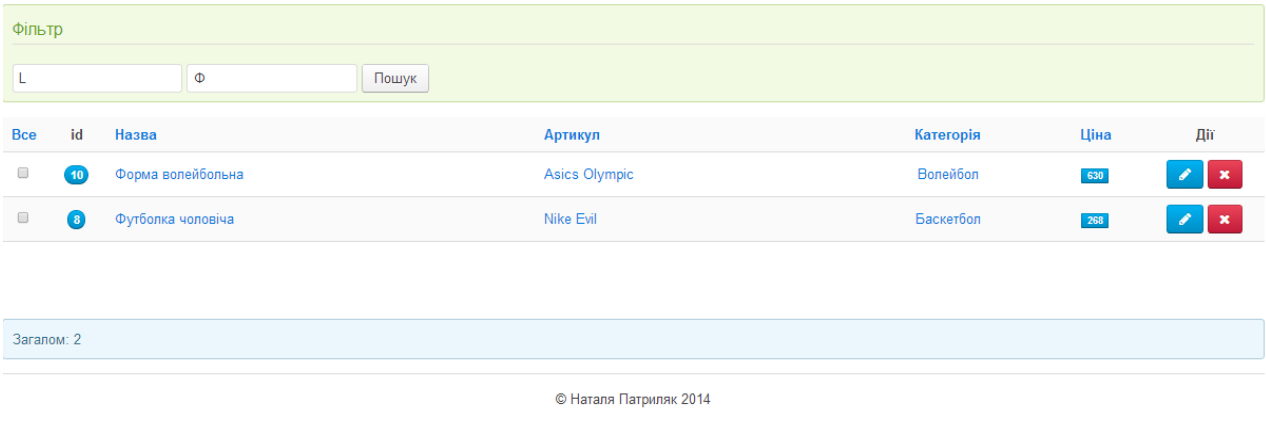

Рисунок 3.11– Пошук за відповідним товаром

3.8. Апаратні засоби комп’ютера

Для розробленого програмного забезпечення пропонується використовувати персональні комп’ютери або ноутбуки з приблизними характеристиками:

- Екран:15.6" (1366x768) WXGA HD
- Процесор: Intel Core i5-520M (2.4 ГГц)
- Об’єм оперативної пам’яті: 4 ГБ
- Тип оперативної пам’яті: DDR3
- Чіпсет: Intel HM55
- Графічний адаптер: NVIDIA GeForce GT330M, 1 ГБ
- Об’єм HDD: 500 ГБ

3.9. Програмні операційні ресурси

Програмне забезпечення буває двох типів: системне та прикладне. Системне програмне забезпечення призначене для підтримки функціонування комп’ютера, для розробки і використання програмних продуктів і створення користувачеві зручних умов роботи. Сюди відносять: операційні системи, інструментальні системи, сервісні програми (програми обслуговування дисків, програми-архіватори, антивірусні програми). Прикладне програмне забезпечення використовують для вирішення типових практичних завдань, для обробки алгоритмів, графічних зображень. До нього належать прикладні програми загального і спеціального призначення. Сюди відносять: текстові редактори, видавничі системи, електронні таблиці, системи керування базами даних, системи автоматизованого проектування, навчальні системи, системи програмування, ігрові програми, системи штучного інтелекту, експертні системи, комунікаційне мережеве програмне забезпечення.

Необхідне програмне забезпечення:

Операційна система: Microsoft Windows 7 (Windows 8)

Антивірусна програма: avast! Antivirus;

Архіватор: WinRAR

Офісний пакет: Microsoft Office 2012

Програмне середовище phpDesigner 8

Denwer(AppServ, HAMPP)

Internet Browser (Mozila FireFox, Opera, IE, Google Chrome)

СКБД MySQL

notepad++.

Програмне забезпечення повинне відповідати інформаційній базі і потребам використання у розроблені програмного застосунку та таким вимогам, як ліцензованість та повнота охоплення функцій.

ВИСНОВКИ

У дипломній роботі був розроблено Web-орієнтована інформаційна система аналізу роботи мережі спортивних магазинів на мові програмування PHP із доступом до реляційної бази даних MySQL через мови доступу до бази даних SQL. Інформаційна система була розроблена у вигляді сайту з елементами JavaScripts та за допомогою UIkit - модульный CSS фреймворк веб-интерфейса.

Результатом дипломної роботи є створення інформаційної системи для мережі спортивних магазинів. Система досволяє відслідковувати весь рух товару у мережі магазинів від поступлення на склад до продажі.

Система забезпечує максимально зручний інтерфейс, мінімальну ціну імплементації, та високу швидкодію.

У дипломній роботі:

- Проведено дослідження предметної області – сфери послуг.
- Проведено огляд наявних інформаційних систем.
- Сформульовано мету розроблення та описано призначення інформаційної системи, побудовано дерево цілей.
- Проведено аналіз технологій та засобів розробки інформаційної системи та вибрані засоби розв’язання поставленої задачі.
- Розроблено інформаційну систему.
- Продемонстровано роботу інформаційної системи.

В інформаційній системі ми маємо змогу редагувати дані в особистому кабінеті, підключати нові модулі, чи редагувати вже існуючі, надавати права доступу користувачам, переглядати, додавати, редагувати, видаляти дані про існуючі товари, про товари на складі, наявні в магазинах, про виробників та підрозділи мережі, слідкувати за продажами.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Берко А.Ю., Висоцька В.А. Проектування навігаційного графу web-сторінок бази даних систем електронної комерції // Вісник НУ “Львівська політехніка”. Комп’ютерні науки та інформаційні технології.– Львів, № 521. –2004.– С.48–57.

2. Берко А.Ю., Висоцька В.А., Чирун Л.В. Алгоритми опрацювання інформаційних ресурсів у системах електронної комерції // Вісник НУ “Львівська політехніка”. Інформаційні системи та мережі. Львів, № 519.– 2004. – С.10–20.

3. Бондаренко А.О. Електронний бізнес та електронна комерція. Львів: НУ “Львівська політехніка”. 2020.– С.330.

4. Гончаренко О.М. Основи електронної комерції. Видавництво Київ: КНЕУ. 2017. – С. 290.

5. Коваленко І.М. Інформаційні технології в електронній комерції. Київ: Київський національний університет. 2016. – С. 320.

6. Ковальчук С.А. Розробка веб-додатків для електронної комерції. Харків: Харківський національний університет. 2018.– С. 310.

7. Кісь, Я.П., Іванік І.В. Реалізація інтернет-магазину з використанням інтелектуальної компоненти // Вісник. НУ "Львівська політехніка". Львів, № 673.– 2014. – С.121-127.

8. Манзій О.С., Тесак І.Є., Кавалець І.І., Чарковська Н.В. Дискретна мтематика. Практикум: нав.посібник // Львів: Видавництво Львівська політехніка. 2016.– С.212.

9. Нікольський Ю.В., Пасічник В.В., Щербина Ю.М. Дискретна математика. Підручник. – Львів : “Магнолія Плюс”, 2005. – С. 608.

10. Підлісний В.М., Ліпич Л.М. Електронна комерція: навчальний посібник. Луцьк: Видавництво ЛНТУ 2015. – С.280.

11. Поліщук В.П. Інформаційні системи в електронній комерції. Вінниця: Видавництво ВНТУ 2019. – С. 270.

12. Романенко Н.І. Електронна комерція та веб-технології. Київ: Видавництво НТУУ "КПІ" 2022. – С. 300.
13. Савченко М.В. Технології веб-розробки для інтернет-магазинів. Одеський національний університет. 2021. – С.295.
14. Bret Williams, Jonathan Bownds. *Mastering Magento* 2 - Second Edition. Packt Publishing 2018. – С. 506.
15. Mohan P. Desai . Designing Web-Based Applications for E-Commerce. CRC Press 2014. – С. 290.
16. Max Amillion. The Art of E-commerce: Building and Designing a Successful Online Store. Digital Publishing House 2019. – С.380.
17. John Smith. E-commerce Website Development Guide. Tech Publishing 2020.– С. 320.
18. Офіційний сайт мови програмування PHP - http://php.net

ДОДАТКИ

База даних

```
SET SQL_MODE="NO_AUTO_VALUE_ON_ZERO";

--
-- БД: `deep_db`
--

-- --------------------------------------------------------

--
-- Структура таблиці `dm_account`
--

CREATE TABLE `dm_account` (
  `id` int(20) NOT NULL auto_increment,
  `name` varchar(255) default NULL,
  `username` varchar(35) NOT NULL,
  `password` varchar(35) NOT NULL,
  `contact_phone` varchar(255) default NULL,
  `contact_email` varchar(255) default NULL,
  `active` tinyint(2) NOT NULL default '1',
  PRIMARY KEY  (`id`)
) ENGINE=MyISAM  DEFAULT CHARSET=utf8 AUTO_INCREMENT=11 ;

--
-- Дамп даних таблиці `dm_account`
--

INSERT INTO `dm_account` VALUES (1, 'root', 'root', 'root', NULL, NULL, 1);
INSERT INTO `dm_account` VALUES (10, 'test', 'test', 'test', '123', '123@123.com', 0);

-- --------------------------------------------------------

--
-- Структура таблиці `dm_account_access`
--

CREATE TABLE `dm_account_access` (
  `id` int(255) NOT NULL auto_increment,
  `id_account` int(255) NOT NULL,
  `id_module` int(255) NOT NULL,
  PRIMARY KEY  (`id`)
) ENGINE=MyISAM  DEFAULT CHARSET=utf8 AUTO_INCREMENT=3 ;

--
-- Дамп даних таблиці `dm_account_access`
--


-- --------------------------------------------------------

--
```

```
-- Структура таблиці `dm_modules`
--

CREATE TABLE `dm_modules` (
  `id` int(255) NOT NULL auto_increment,
  `id_parent` int(255) default '0',
  `module` varchar(255) character set utf8 default NULL,
  `workspace` varchar(255) default NULL,
  `coreclass` varchar(255) default NULL,
  `name` varchar(255) character set utf8 default NULL,
  `order` int(255) default '0',
  PRIMARY KEY  (`id`)
) ENGINE=MyISAM  DEFAULT CHARSET=latin1 AUTO_INCREMENT=29 ;

--
-- Дамп даних таблиці `dm_modules`
--

INSERT INTO `dm_modules` VALUES (4, 0, 'account', 'account', 'account.class.php', 'Кабінет', 10);
INSERT INTO `dm_modules` VALUES (24, 0, 'stock', 'catalog', 'stock.class.php', 'Склад', 42);
INSERT INTO `dm_modules` VALUES (25, 0, 'store', 'catalog', 'store.class.php', 'Магазини', 44);
INSERT INTO `dm_modules` VALUES (26, 0, 'goods', 'catalog', 'goods.class.php', 'Товари', 45);
INSERT INTO `dm_modules` VALUES (27, 0, 'link', 'catalog', 'link.class.php', 'Наявність', 46);
INSERT INTO `dm_modules` VALUES (22, 0, 'categories', 'catalog', 'categories.class.php', 'Категорії', 40);
INSERT INTO `dm_modules` VALUES (23, 0, 'vendor', 'catalog', 'vendor.class.php', 'Виробники', 41);
INSERT INTO `dm_modules` VALUES (1, 0, 'manager', 'manager', 'manager.class.php', 'Модулі', 20);
INSERT INTO `dm_modules` VALUES (28, 0, 'sales', 'catalog', 'sales.class.php', 'Продажі', 47);
INSERT INTO `dm_modules` VALUES (20, 0, 'administrators', 'manager', 'administrators.class.php',
'Адміністратори', 35);
INSERT INTO `dm_modules` VALUES (21, 20, 'administrators_access', 'manager',
'administrators_access.class.php', 'Права доступу', 36);

-- --------------------------------------------------------

--
-- Структура таблиці `dm_savesort`
--

CREATE TABLE `dm_savesort` (
  `id` int(255) NOT NULL auto_increment,
  `id_module` int(255) NOT NULL,
  `field` varchar(255) character set utf8 NOT NULL,
  `ascending` tinyint(4) NOT NULL default '1',
  PRIMARY KEY  (`id`)
) ENGINE=MyISAM  DEFAULT CHARSET=latin1 AUTO_INCREMENT=15 ;

--
-- Дамп даних таблиці `dm_savesort`
--

INSERT INTO `dm_savesort` VALUES (13, 3, 'id', 1);
INSERT INTO `dm_savesort` VALUES (10, 7, 'data', 1);
INSERT INTO `dm_savesort` VALUES (14, 26, 'code', 1);

-- --------------------------------------------------------

--
-- Структура таблиці `dp_categories`
```

```
--

CREATE TABLE `dp_categories` (
  `id` int(255) NOT NULL auto_increment,
  `name` varchar(255) character set utf8 default NULL,
  PRIMARY KEY  (`id`)
) ENGINE=MyISAM  DEFAULT CHARSET=latin1 AUTO_INCREMENT=17 ;

--
-- Дамп даних таблиці `dp_categories`
--

INSERT INTO `dp_categories` VALUES (2, 'Футбол');
INSERT INTO `dp_categories` VALUES (3, 'Баскетбол');
INSERT INTO `dp_categories` VALUES (4, 'Волейбол');
INSERT INTO `dp_categories` VALUES (5, 'Великий теніс');
INSERT INTO `dp_categories` VALUES (6, 'Настільний теніс');
INSERT INTO `dp_categories` VALUES (7, 'Бадмінтон');
INSERT INTO `dp_categories` VALUES (8, 'Біг');
INSERT INTO `dp_categories` VALUES (9, 'Лижі бігові');
INSERT INTO `dp_categories` VALUES (10, 'Сноуборд');
INSERT INTO `dp_categories` VALUES (11, 'Фітнес');
INSERT INTO `dp_categories` VALUES (12, 'Тренажери');
INSERT INTO `dp_categories` VALUES (13, 'Важка атлетика');
INSERT INTO `dp_categories` VALUES (14, 'Єдиноборства');
INSERT INTO `dp_categories` VALUES (15, 'Гімнастика');
INSERT INTO `dp_categories` VALUES (16, 'Плавання');

-- --------------------------------------------------------

--
-- Структура таблиці `dp_goods`
--

CREATE TABLE `dp_goods` (
  `id` int(255) NOT NULL auto_increment,
  `code` varchar(255) character set utf8 default NULL,
  `name` varchar(255) character set utf8 NOT NULL,
  `id_category` int(255) NOT NULL default '0',
  `id_vendor` int(255) NOT NULL default '0',
  `price` varchar(255) character set utf8 NOT NULL default '0.0',
  PRIMARY KEY  (`id`)
) ENGINE=MyISAM  DEFAULT CHARSET=latin1 AUTO_INCREMENT=27 ;

--
-- Дамп даних таблиці `dp_goods`
--

INSERT INTO `dp_goods` VALUES (1, 'ФМ-500', 'Футбольный мяч', 2, 0, '1000');
INSERT INTO `dp_goods` VALUES (2, 'Чемпион', 'М\\"яч сувенірний', 2, 0, '46');
INSERT INTO `dp_goods` VALUES (3, 'Nike Sqad', 'Футболка чоловіча', 2, 0, '324');
INSERT INTO `dp_goods` VALUES (4, 'Nike Tiempo', 'Бутси для залу', 2, 0, '1050');
INSERT INTO `dp_goods` VALUES (5, 'Adidas Tiro', 'Сумка чоловіча', 2, 0, '456');
INSERT INTO `dp_goods` VALUES (6, 'Nike Dual', 'Кросівки чоловічі', 3, 0, '863');
INSERT INTO `dp_goods` VALUES (7, 'Start Up', 'М\\"яч баскетбольний', 3, 0, '270');
INSERT INTO `dp_goods` VALUES (8, 'Nike Evil', 'Футболка чоловіча', 3, 0, '268');
INSERT INTO `dp_goods` VALUES (9, 'Mikasa MVA12', 'Мяч волейбольний', 4, 0, '420');
INSERT INTO `dp_goods` VALUES (10, 'Asics Olympic', 'Форма волейбольна', 4, 0, '630');
```

```
INSERT INTO `dp_goods` VALUES (11, 'Mizuno Twister2', 'Кросівки жіночі', 4, 0, '1000');
INSERT INTO `dp_goods` VALUES (12, 'Nike Maria2', 'Плаття жіноче', 5, 0, '520');
INSERT INTO `dp_goods` VALUES (13, 'Start Up advanse5', 'Ракетка для н.т.', 6, 0, '295');
INSERT INTO `dp_goods` VALUES (14, 'Enabe Game', 'Стіл для н.т.', 6, 0, '6500');
INSERT INTO `dp_goods` VALUES (15, 'Larsen Alum 308', 'Набір для бадмінтона', 7, 0, '125');
INSERT INTO `dp_goods` VALUES (16, 'A5-529', 'Капрі жіночі', 8, 0, '290');
INSERT INTO `dp_goods` VALUES (17, 'Larsen Life NN75', 'Комплект лижний', 9, 0, '980');
INSERT INTO `dp_goods` VALUES (18, 'Monte Grande', 'Штани гірськолижні', 10, 0, '450');
INSERT INTO `dp_goods` VALUES (19, 'Iron Body', 'Батут', 11, 0, '860');
INSERT INTO `dp_goods` VALUES (20, 'Start Up DB', 'Турнік в проріз', 11, 0, '765');
INSERT INTO `dp_goods` VALUES (21, 'Larsen Tm5', 'Дорожка електрична', 12, 0, '2500');
INSERT INTO `dp_goods` VALUES (22, 'Brumer GUM32', 'Тренажер силовий', 13, 0, '5000');
INSERT INTO `dp_goods` VALUES (23, 'Gramer EM32', 'Підставка під штангу', 13, 0, '1020');
INSERT INTO `dp_goods` VALUES (24, 'Рэй спорт', 'Перчатки', 14, 0, '658');
INSERT INTO `dp_goods` VALUES (25, 'AB 283', 'Скакалка', 15, 0, '140');
INSERT INTO `dp_goods` VALUES (26, 'Larsen 42', 'Шапочка', 16, 0, '100');

-- --------------------------------------------------------

--
-- Структура таблиці `dp_link`
--

CREATE TABLE `dp_link` (
  `id` int(255) NOT NULL auto_increment,
  `id_good` int(255) NOT NULL,
  `id_store` int(255) NOT NULL,
  `count` int(255) NOT NULL,
  PRIMARY KEY  (`id`)
) ENGINE=MyISAM  DEFAULT CHARSET=latin1 AUTO_INCREMENT=14 ;

--
-- Дамп даних таблиці `dp_link`
--

INSERT INTO `dp_link` VALUES (1, 1, 1, 5);
INSERT INTO `dp_link` VALUES (3, 17, 1, 3);
INSERT INTO `dp_link` VALUES (4, 15, 1, 4);
INSERT INTO `dp_link` VALUES (5, 11, 2, 2);
INSERT INTO `dp_link` VALUES (6, 10, 2, 3);
INSERT INTO `dp_link` VALUES (7, 4, 3, 2);
INSERT INTO `dp_link` VALUES (8, 7, 3, 3);
INSERT INTO `dp_link` VALUES (9, 11, 3, 3);
INSERT INTO `dp_link` VALUES (10, 24, 3, 3);
INSERT INTO `dp_link` VALUES (11, 8, 3, 4);
INSERT INTO `dp_link` VALUES (12, 5, 2, 5);
INSERT INTO `dp_link` VALUES (13, 8, 2, 5);

-- --------------------------------------------------------

--
-- Структура таблиці `dp_sales`
--

CREATE TABLE `dp_sales` (
  `id` int(255) NOT NULL auto_increment,
  `id_good` int(255) NOT NULL,
  `id_store` int(255) NOT NULL,
```

```
  `count` int(255) NOT NULL,
  `datetime` datetime default NULL,
  PRIMARY KEY  (`id`)
) ENGINE=MyISAM  DEFAULT CHARSET=latin1 AUTO_INCREMENT=6 ;

--
-- Дамп даних таблиці `dp_sales`
--

INSERT INTO `dp_sales` VALUES (1, 1, 1, 1, '2014-06-25 05:45:00');
INSERT INTO `dp_sales` VALUES (2, 15, 1, 1, '2014-06-25 10:30:00');
INSERT INTO `dp_sales` VALUES (3, 8, 2, 1, '2014-05-28 12:15:00');
INSERT INTO `dp_sales` VALUES (4, 11, 3, 4, '2014-05-30 12:15:00');
INSERT INTO `dp_sales` VALUES (5, 8, 2, 4, '2014-05-30 12:15:00');

-- --------------------------------------------------------

--
-- Структура таблиці `dp_stock`
--

CREATE TABLE `dp_stock` (
  `id` int(255) NOT NULL auto_increment,
  `id_good` int(255) NOT NULL,
  `count` int(255) NOT NULL default '0',
  PRIMARY KEY  (`id`)
) ENGINE=MyISAM  DEFAULT CHARSET=latin1 AUTO_INCREMENT=23 ;

--
-- Дамп даних таблиці `dp_stock`
--

INSERT INTO `dp_stock` VALUES (1, 1, 80);
INSERT INTO `dp_stock` VALUES (2, 8, 16);
INSERT INTO `dp_stock` VALUES (3, 12, 35);
INSERT INTO `dp_stock` VALUES (4, 18, 15);
INSERT INTO `dp_stock` VALUES (5, 19, 5);
INSERT INTO `dp_stock` VALUES (6, 14, 10);
INSERT INTO `dp_stock` VALUES (7, 13, 70);
INSERT INTO `dp_stock` VALUES (8, 6, 35);
INSERT INTO `dp_stock` VALUES (9, 9, 25);
INSERT INTO `dp_stock` VALUES (10, 7, 33);
INSERT INTO `dp_stock` VALUES (11, 15, 32);
INSERT INTO `dp_stock` VALUES (12, 17, 32);
INSERT INTO `dp_stock` VALUES (13, 26, 60);
INSERT INTO `dp_stock` VALUES (14, 25, 35);
INSERT INTO `dp_stock` VALUES (15, 24, 27);
INSERT INTO `dp_stock` VALUES (16, 23, 30);
INSERT INTO `dp_stock` VALUES (17, 22, 20);
INSERT INTO `dp_stock` VALUES (18, 21, 35);
INSERT INTO `dp_stock` VALUES (19, 20, 30);
INSERT INTO `dp_stock` VALUES (20, 20, 30);
INSERT INTO `dp_stock` VALUES (21, 19, 36);

-- --------------------------------------------------------

--
-- Структура таблиці `dp_store`
```

```
--

CREATE TABLE `dp_store` (
  `id` int(255) NOT NULL auto_increment,
  `name` varchar(255) character set utf8 NOT NULL,
  `telphone` varchar(255) character set utf8 NOT NULL,
  `adress` varchar(255) character set utf8 NOT NULL,
  PRIMARY KEY  (`id`)
) ENGINE=MyISAM  DEFAULT CHARSET=latin1 AUTO_INCREMENT=4 ;

--
-- Дамп даних таблиці `dp_store`
--

INSERT INTO `dp_store` VALUES (1, 'ТЦ Омега', '54654', 'м.Київ, вул.Шептицького 2');
INSERT INTO `dp_store` VALUES (2, 'ТЦ Армада', '54654984', 'м.Орск, пр.Горького, 45');
INSERT INTO `dp_store` VALUES (3, 'ТСК Бирь', '5456', 'м.Львів, вул.Природна 19');

-- --------------------------------------------------------

--
-- Структура таблиці `dp_vendor`
--

CREATE TABLE `dp_vendor` (
  `id` int(255) NOT NULL auto_increment,
  `name` varchar(255) character set utf8 default NULL,
  `subvendor` varchar(255) character set utf8 default NULL,
  `adress` varchar(255) character set utf8 default NULL,
  `email` varchar(255) character set utf8 default NULL,
  `telphone` varchar(255) character set utf8 default NULL,
  PRIMARY KEY  (`id`)
) ENGINE=MyISAM  DEFAULT CHARSET=latin1 AUTO_INCREMENT=6 ;

--
-- Дамп даних таблиці `dp_vendor`
--

INSERT INTO `dp_vendor` VALUES (1, 'Adidas', '', 'м.Мюнхен, вул.Бандери, 15', 'adidas@adidas.com',
'12345');
INSERT INTO `dp_vendor` VALUES (2, 'Nike', '', 'м.Київ, вул.Шевченка, 72', 'nike@nike.com',
'56421564');
INSERT INTO `dp_vendor` VALUES (3, 'Bercuda', '', 'м.Харків, вул.Ленінська, 23',
'Bercuda@bercuda.com', '5464675');
INSERT INTO `dp_vendor` VALUES (4, 'Lanser', '', 'м. Одеса, вул.Київська, 26', 'Lanser@lanser.com',
'6457876');
INSERT INTO `dp_vendor` VALUES (5, 'Alonsa', '', 'м.Київ, вул.Природна, 57', 'Alonsa@alonsa.com',
'745765464');
```

Інтернет файли

Index.php

```
 <?
session_start();
ERROR_REPORTING(E_ALL);
DEFINE('WVERSION', 'ROOT');
```

```
require_once('conf.php');
date_default_timezone_set('Europe/Moscow');

require_once 'classes/common.class.php';
$wcommon = new wcommon();
$modules_array = $wcommon->modules_loader(true);

function __autoload($class)
{
	global $modules_array;
	if (isset($modules_array[$class]))
	{
		require_once $modules_array[$class];
	}
	else
		return false;
}
$authorize = $wcommon->authorize();
echo $wcommon->outRoute();
?>
```

styles.css

```
 .form_buildLetter {
	max-width: 800px;
	padding: 19px 29px 29px;
	margin: 0 auto 20px;
	background-color: #F4F4F4;
	border: 1px solid #e5e5e5;
      -webkit-border-radius: 5px;
         -moz-border-radius: 5px;
	    border-radius: 5px;
      -webkit-box-shadow: 0 1px 2px rgba(0,0,0,.05);
         -moz-box-shadow: 0 1px 2px rgba(0,0,0,.05);
	    box-shadow: 0 1px 2px rgba(0,0,0,.05);
}

.editor-visual {
	margin-bottom: 15px !important;
	margin-top: 15px !important;
}

.h2-special {
	text-align:Center;
	font-family: Arial;
	letter-spacing: -1px;
	color: #CFD1CF;
	text-shadow: 0 1px 0 #fff;
	font-weight:normal;
}
```

Login.htm

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Авторизуемся</title>
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<link href="./js/bootstrap/css/bootstrap.css" rel="stylesheet">
```

```
<link href="./js/bootstrap/css/bootstrap-responsive.css" rel="stylesheet">
<link href="./assets/style.css" rel="stylesheet">
   <style type="text/css">
     body {
       padding-top: 40px;
       padding-bottom: 40px;
       background-color: #f5f5f5;
     }

     .form-signin {
       max-width: 300px;
       padding: 19px 29px 29px;
       margin: 0 auto 20px;
       background-color: #fff;
       border: 1px solid #e5e5e5;
       -webkit-border-radius: 5px;
          -moz-border-radius: 5px;
               border-radius: 5px;
       -webkit-box-shadow: 0 1px 2px rgba(0,0,0,.05);
          -moz-box-shadow: 0 1px 2px rgba(0,0,0,.05);
               box-shadow: 0 1px 2px rgba(0,0,0,.05);
     }
     .form-signin .form-signin-heading,
     .form-signin .checkbox {
       margin-bottom: 10px;
     }
     .form-signin input[type="text"],
     .form-signin input[type="password"] {
       font-size: 16px;
       height: auto;
       margin-bottom: 15px;
       padding: 7px 9px;
     }

   </style>
</head>
<body>
<div class="container-fluid">
   <div class="container">
     <form class="form-signin" action="./index.php" method="post">
       <h2 class="form-signin-heading">WebTrix ROOT2</h2>
       <input name="ulogin" type="text" class="input-block-level" placeholder="Вводим логин" required>
       <input name="upassword" type="password" class="input-block-level" placeholder="Вводим пароль"
required>
       <button class="btn btn-large btn-primary" type="submit">Войти</button>
     </form>
   </div>
         <footer style="text-align:center;">
                 <p>&copy; WebTrix Dynamic Tools 2014</p>
         </footer>
</div>

</body>
</html>
```

account.class.php

```
<?
         class account extends wcommon
```

```
{
	private static $db_table = 'dm_account';
	private static $mod_name = 'Личный кабинет';

	function __construct(){}

	public function init()
	{
		if(isset($_POST['formEdit_account']))
			$this->account_editData();

		$result = $this->account_info();
		$data = '
		<h1 style="text-align:center;"><a href="#">'.self::$mod_name.'</a></h1>
		' . parent::breadway() . '
		<div class="row-fluid">
			<div class="span12">
				'.$result.'
			</div>
		</div>';

		$template = file_get_contents('assets/output.tpl');
		$template = str_replace('{DATA}', $data, $template);
		$template = str_replace('{TITLE}', self::$mod_name, $template);
		$template = str_replace('{MENU}', parent::head_menu(), $template);
		return $template;
	}

	private function account_info()
	{
		$uac = $_SESSION['uac'];
		$data_sql = parent::dbc()->query("SELECT * FROM `".self::$db_table."` WHERE
`id` = '".$uac."'");
		$data_row = $data_sql->fetch_assoc();
		$editForm = '
		<div class="uk-vertical-align uk-text-center uk-height-1-1">
			<div class="uk-vertical-align-middle" style="width: 250px;">
				<img class="uk-margin-bottom" width="140" height="120"
src=" " alt="">
				<form class="uk-panel uk-panel-box uk-form"
action="'.ENGINE_URL.'account" method="post">
					<div class="uk-form-row">
						<input class="uk-width-1-1 uk-form-large"
type="text" placeholder="Логин" value="'.$data_row['username'].'" disabled>
					</div>
					<div class="uk-form-row">
						<input class="uk-width-1-1 uk-form-large"
type="password" name="inputPassword" id="inputPassword" placeholder="Пароль"
value="'.$data_row['password'].'" required>
					</div>
					<div class="uk-form-row">
						<input type="submit" class="uk-width-1-1 uk-button
uk-button-primary uk-button-large" name="formEdit_account" value="Сохранить">
					</div>
				</form>
			</div>
		</div>';
		return $editForm;
```

```
		}

		private function account_editData()
		{
			$uac = $_SESSION['uac'];
			$save_pw		= $_POST['inputPassword'];

			$query = mysql_query("UPDATE `".self::$db_table."` SET
				`password`	= '".$save_pw."'
			WHERE `id` = '".$uac."'");
			return;
		}
	}
?>
```

categories.class.php

```
<?
	class categories extends wcommon
	{
		private static $db_table = 'dp_categories';
		private static $mod_name = 'Категории';
		private static $parent_field = false;

		public  static $url_name = 'categories';
		private $db_fields = array();

		function __construct()
		{
			$this->db_fields = array(
				array(
					'type'			=> 'varchar',
					'field'			=> 'name',
					'placeholder'	=> 'Название',
					'listing'		=> true,
					'caption'		=> 'Название',
					'align'			=> 'left',
					'width'			=> false,
					'customHref'	=> false,
				),
			);
		}

		private function inRoute()
		{
			if(isset($_GET['build']))
				return parent::object_editForm(false, self::$db_table, $this->db_fields,
self::$parent_field, self::$url_name);

			if(isset($_GET['edit']))
				return parent::object_editForm($_GET['edit'], self::$db_table, $this->db_fields,
self::$parent_field, self::$url_name);

			if(isset($_GET['remove']))
				return parent::object_removeData(self::$db_table, $_GET['remove']);

			if(isset($_POST['data']))
			{
				switch($_POST['data'])
				{
					case 'objectInsert':
						return parent::object_InsertData(self::$db_table, $this->db_fields);
```

```
                                        break;
                                }
                        }
                }

                public function init()
                {
                        if(isset($_GET) || isset($_POST))
                                if(isset($_GET['jump']) && $_GET['jump'] == true) return $this->inRoute(); else
$after = $this->inRoute();

                        $condition  = (isset($_GET['parent']) && !empty($_GET['parent']) &&
(isset(self::$parent_field) && self::$parent_field != false)) ? " WHERE `".self::$parent_field."` =
'".intval($_GET['parent'])."' " : '';
                        $build_url  = (isset($_GET['parent']) && !empty($_GET['parent']) &&
(isset(self::$parent_field) && self::$parent_field != false)) ? ENGINE_URL .
self::$url_name.'?build=true&parent='.intval($_GET['parent']) : ENGINE_URL . self::$url_name.'?build=true';
                        $build_url .= '&rewind='.base64_encode($_SERVER['REQUEST_URI']);
                        $result = (isset($after)) ? $after : parent::object_listE(self::$db_table, $this->db_fields,
self::$url_name, $condition);
                        return parent::template_draw($result, self::$mod_name, $build_url);
                }
        }
?>
```

goods.class.php

```
<?
        class goods extends wcommon
        {
                private static $db_table = 'dp_goods';
                private static $mod_name = 'Товары';
                private static $parent_field = false;

                public  static $url_name = 'goods';
                private $db_fields = array();

                function __construct()
                {
                        $this->db_fields = array(

                                array(
                                        'type'                  => 'varchar',
                                        'field'                 => 'name',
                                        'placeholder'   => 'Название',
                                        'listing'               => true,
                                        'caption'               => 'Название',
                                        'align'                 => 'left',
                                        'width'                 => false,
                                        'customHref'    => false,
                                ),
                                array(
                                        'type'                  => 'varchar',
                                        'field'                 => 'code',
                                        'placeholder'   => 'Артикул',
                                        'listing'               => true,
                                        'caption'               => 'Артикул',
                                        'align'                 => 'left',
                                        'width'                 => false,
                                        'customHref'    => false,
                                ),
                                array(
                                        'type'                  => 'select',
                                        'field'                 => 'id_category',
                                        'placeholder'   => 'Категория',
```

```
                                        'list'                  => parent::IDN_LIST('dp_categories'),
                                        'listing'       => true,
                                        'caption'               => 'Категория',
                                        'align'         => 'center',
                                        'width'         => '190',
                                        'customHref'    => false,
                                ),
                                array(
                                        'type'                  => 'intval',
                                        'field'                 => 'price',
                                        'placeholder'   => 'Цена',
                                        'listing'       => true,
                                        'caption'               => 'Цена',
                                        'align'         => 'center',
                                        'width'         => '90',
                                        'customHref'    => false,
                                ),
                        );
                }

                private function inRoute()
                {
                        if(isset($_GET['build']))
                                return parent::object_editForm(false, self::$db_table, $this->db_fields,
self::$parent_field, self::$url_name);

                        if(isset($_GET['edit']))
                                return parent::object_editForm($_GET['edit'], self::$db_table, $this->db_fields,
self::$parent_field, self::$url_name);

                        if(isset($_GET['remove']))
                                return parent::object_removeData(self::$db_table, $_GET['remove']);

                        if(isset($_POST['data']))
                        {
                                switch($_POST['data'])
                                {
                                        case 'objectInsert':
                                                return parent::object_InsertData(self::$db_table, $this->db_fields);
                                        break;
                                }
                        }
                }

                public function init()
                {
                        if(isset($_GET) || isset($_POST))
                                if(isset($_GET['jump']) && $_GET['jump'] == true) return $this->inRoute(); else
$after = $this->inRoute();

                        $condition  = (isset($_GET['parent']) && !empty($_GET['parent']) &&
(isset(self::$parent_field) && self::$parent_field != false)) ? " WHERE `".self::$parent_field."` =
'".intval($_GET['parent'])."' " : '';
                        $build_url  = (isset($_GET['parent']) && !empty($_GET['parent']) &&
(isset(self::$parent_field) && self::$parent_field != false)) ? ENGINE_URL .
self::$url_name.'?build=true&parent='.intval($_GET['parent']) : ENGINE_URL . self::$url_name.'?build=true';
                        $build_url .= '&rewind='.base64_encode($_SERVER['REQUEST_URI']);

                        $result = (isset($after)) ? $after : parent::object_listE(self::$db_table, $this->db_fields,
self::$url_name, $condition);


                        return parent::template_draw($result, self::$mod_name, $build_url);
                }
```

```
	}
?>
```

link.class.php

```
<?
	class link extends wcommon
	{
		private static $db_table = 'dp_link';
		private static $mod_name = 'Наличие товара';
		private static $parent_field = false;

		public  static $url_name = 'link';
		private $db_fields = array();

		function __construct()
		{
			$this->db_fields = array(
				array(
					'type'			=> 'select',
					'field'			=> 'id_good',
					'placeholder'	=> 'Товар',
					'list'			=> parent::IDN_LIST('dp_goods'),
					'listing'		=> true,
					'caption'		=> 'Товар',
					'align'			=> 'left',
					'width'			=> false,
					'customHref'	=> false,
				),
				array(
					'type'			=> 'select',
					'field'			=> 'id_store',
					'placeholder'	=> 'Магазин',
					'list'			=> parent::IDN_LIST('dp_store'),
					'listing'		=> true,
					'caption'		=> 'Магазин',
					'align'			=> 'left',
					'width'			=> false,
					'customHref'	=> false,
				),


				array(
					'type'			=> 'intval',
					'field'			=> 'count',
					'placeholder'	=> 'Количество',
					'listing'		=> true,
					'caption'		=> 'Количество',
					'align'			=> 'center',
					'width'			=> '90',
					'customHref'	=> false,
				),
			);
		}

		private function spent_good($id_good, $count)
		{
			$query = parent::dbc()->query("UPDATE `dp_stock` SET `count` = count-".intval($count)."
WHERE `id_good` = '".intval($id_good)."'");
			return;
		}
```