Міністерство освіти і науки України
Сумський державний університет
Навчально-науковий інститут бізнес-технологій «УАБС»
Кафедра економічної кібернетики

КВАЛІФІКАЦІЙНА МАГІСТЕРСЬКА РОБОТА

на тему «Автоматизація обліку та реалізації товарів на підприємстві»

Виконав студент 2 курсу, групи ЕК.м-61а
(номер курсу) (шифр групи)
Спеціальності 051«Економіка («Економічна кібернетика»)»
Братина Я.О.
(прізвище, ініціали студента)
Керівник д.е.н., доцент Олійник В.М.
(посада, науковий ступінь, прізвище, ініціали)

Суми – 2018

РЕФЕРАТ

Дипломна робота: 146 с., 6 табл., 26 рис., 2 додатка, 35 джерел. Об’єктом дослідження є діяльність підприємства ПП «Кулішенко Н.А»
в сфері обліку та реалізації товарів.

Предмет – автоматизація механізму продажу та обліку товарів на підприємстві.

Метою є створення веб – сайту для забезпечення ведення обліку товарів та продажів товару в мережі інтернет. Це допоможе вирішити проблеми використання паперових носіїв інформації, та презведе до скорочення витрат і часу для продажу та обліку товарів.

Цілями є створення і впровадження прототипу програмного забезпечення автоматизації механізму продажу та обліку товарів на підприємстві.

Сформульовано наступні висновки: в ході дипломної роботи було створено прототип автоматизації механізму продажу та обліку товарів на підприємстві. Основні напрямки автоматизації є створення веб - сайту, який спрощує ведення обліку товарів та надає можливість покупки товарів підприємства у зручному місці та в зручний час. Було розроблено інструкції по використанню автоматизованої системи як для працівників підприємства, так і для клієнтів.

КЛЮЧОВІ СЛОВА: АВТОМАТИЗАЦІЯ, HTML, PHP, SUBLIME TEXT, OPENSERVER, CSS.

ЗМІСТ

ВСТУП

На сучасному етапі розвитку важко уявити підприємство без використання інформаційних технологій. Вони проникли майже у всі сфери підприємницької діяльності, у тому числі і у роздрібну торгівлю.

Необхідність автоматизації механізму клієнтського обслуговування на підприємстві, використовуючи новітні технології, вкрай бажана. Сучасний бізнес стає усе більш динамічним, боротьба за конкурентні переваги найчастіше перетворюється в погоню за сучасними управлінськими та інформаційними технологіями. До часу впровадження автоматизованої системи на підприємстві використовувались застарілі технології, тому дивлячись на всі переваги автоматизації, було прийнято рішення про впровадження нової системи у роботу підприємства.

З розвитком технологій ми отримали можливість використовувати такі мережі, як Інтернет у автоматизації на підприємстві. За допомогою цієї мережі можливо вести облік складу та реалізацію товарів.

Інтернет магазин - це реалізоване в мережі Інтернет представництво, шляхом створення Web-додатку для продажу товарів і послуг іншим користувачам мережі Інтернет.

Протягом багатьох останніх років йде активний розвиток сфери IT технологій та електронної комерції, що сприяє притоку кількості клієнтів Інтернет – магазинів та збільшенню кількості реалізованого товару із року в рік. Вигода від інтернет магазину обумовлюється більш низьким рівнем витрат у порівнянні зі звичайною торговою точкою, що може зменшити ціни на товари та повисити конкурентоспроможність даного підприємства.

Розробка Інтернет-магазину та його використання є актуальним питанням на сьогоднішній день, оскільки мільйони людей щодня, не виходячи з дому, купують різні товари в електронних магазинах.

Основною перевагою такого способу роздрібного продажу , також є, приймання заказів онлайн цілодобово, що дає змогу збільшити кількість заказів для підприємства, а для покупця – в зручний ,для нього, час купити товар, використовуючи мережу Інтернет.

За допомогою використання веб – сайту для обліку товарів на підприємстві та за допомогою бази даних, з'являється можливість перевіряти наявність товарів також і на стаціонарних точках продажу з використання комп'ютерів та мережі інтернет.

З всього вище сказаного однозначно зрозуміла актуальність питання розробки Інтернет-магазину та ведення обліку з використанням сучасних Internet-технологій.

Мета роботи полягає у аналізі діяльності підприємства, аналізі використовуємих методах автоматизації на підприємстві, визначення пріоритетних напрямків автоматизації.

Новизна роботи полягає в сучасних технологіях автоматизації підприємства за допомогою сайту, а саме мови програмування PHP і мовою розмітки веб – сторінок HTML для розробки Web-сайту та систему управління базами даних MySQL для розробки бази даних Інтернет-магазину. Також, у впровадженні системи що підвищить ефективність підприємства у сфері обліку та реалізації товарів на підприємстві.

Завдання, що ставляться для дослідження:

– розгляд соціально-економічної сутності поставленої задачі, основних теоретичних відомостей;

– збір відомостей про об'єкт автоматизації (визначення складових компонентів, що мають бути включені до розроблюваної системи) та формування вимог до системи;

– розробка проекту автоматизованої системи;

– створення прототипу системи;

– визначення очікуваних ефектів від впровадження системи.

Для вирішення поставлених задач було розроблено веб – сайт з можливостями обслуговування клієнтів та веденням обліку товарів в мережі інтернет, що дозволить мати постійний доступ до продукції підприємства як покупцям, так і працівникам підприємства віддалених від центрального відділення.

Розроблений сайт матиме простий інтерфейс, що дасть змогу використовувати його навіть не досвідчених користувачеві мережі інтернет. Облік та введення товарів виконується за допомогою сайту, тому працівники що не мають опиту праці з базами даних зможуть виконувати прості операції з додаваня, видалення та обслуговування в базі даних.

З часом, при збільшенні клієнтури підприємства, даній системі не буде потрібно модифікація, так як вона с самого початку розрахована на велику кількість покупців. А також, не потрібно вносити змін при збільшенні кількості локальних представництв і розширенні штату персоналу, система вже має налаштування для поширення.

Використання даної системи дасть змогу підприємству постійно нарощувати кількість клієнтів, товару, а також зменшити час на обробку замовлень та ведення обліку товарів.

1. АВТОМАТИЗАЦІЯ ПРОДАЖІВ ТА ОБЛІКУ ТОВАРІВ НА ПІДПРИЄМСТВІ

1.1 Аналіз існуючих систем автоматизації

Автоматизація підприємства – це процес, у якому функції управління і функції контролю, раніше які виконувались людиною, передаються приладам і автоматичним пристроям.

Успішний розвиток будь-якого підприємства неможливо без автоматизації, яка грунтується на використанні передових інформаційних технологій. Причому вона повинна проводитися комплексно.

Комплексна автоматизація підприємства має на увазі впровадження технічного оснащення та відповідного софта, в результаті чого значна частина робіт може проводитися з мінімальним залученням персоналу. Це дозволяє уникати помилок і затримок, обумовлених людським фактором. Основний інструмент автоматизації - комп'ютерна система, розділена на сектори і здатна виконувати безліч процесів одночасно.

В цій роботі буде розглянуто автоматизацію на прикладі торгових точок, а саме декілька ювелірних магазинів мережі ПП «Кулішенко Н.А».

Успішний магазин - це автоматизована робота всього робочого процесу, який відбувається завдяки чітко налагодженій роботі сучасного торговельного обладнання та програмного забезпечення, яке виступає в ролі ефективного автоматичного керування всіма торговими процесами як для супермаркетів, так і для невеликих торгових точок. Для сучасного торгового бізнесу повна автоматизація магазину - це процвітання бізнесу і постійне зростання прибутку.

Автоматизація особливо важлива для бізнесу. Правильна організація поставок товару прямо залежить від аналізу товарообігу, визначення популярних товарів, швидкості складання замовлення та здійснення

розрахунків з постачальниками. Якість та швидкість обслуговування покупця залежать від того, наскільки швидко і точно працюють касове обладнання, торговельні ваги, сканери штрих-кодів, принтери чеків та етикеток.

Автоматизація підприємства допоможе оптимізувати весь торговельний процес, підвищити облікові і маркетингові операції, що підвищують ефективність та працездатність всіх працівників магазину та приверне клієнтів. І в результаті підвищить прибуток, тому Ваша покупка та встановлення обладнання для автоматизації окупиться за мінімальний термін [1].

Під автоматизацією торгівлі можна розуміти програмні і апаратні рішення бюджетного рівня (з початковим функціоналом), спрямовані на забезпечення реалізації різних складових торгового процесу без участі людини. Такими процесами можуть бути:

- розрахунок з покупцями;
- бухгалтерський облік;
- управління товарними запасами;
- керування продажами.

Сучасні магазини і торгові мережі являють собою складну систему складів, що вимагає оперативного обліку продажів і переміщень товарів. Без автоматизації обліку руху товарів і грошей досягти успіху в цьому бізнесі досить складно [8].

Темп сучасного бізнесу не дозволяє відкладати процес реєстрації руху товарів. Більш того, автоматизація в торгівлі важлива не тільки як засіб тотального обліку товару в інтересах власників бізнесу, але, перш за все, в інтересах покупців як засіб поліпшення швидкості і якості обслуговування, звільнення часу продавця, щоб вільно спілкуватися з покупцем.

При організації обліку в торговельній мережі важливу роль відіграють програмні та технічні засоби автоматизації, а так само забезпечення маркування всього товару, який надходить у продаж. Зазвичай тільки частина товару має штрих-кодову маркування, виконану виробником. Для не

марковані товару торговельна мережа повинна сама сформувати штрих-код і прикріпити його до товару. Це можна зробити за допомогою спеціалізованого торгівельного обладнання: принтери етикеток, сканери штрих-кодів, термінали збору даних.

Розглянимо основні переваги автоматизації, які наведені в таблиці 1.1.

Таблиця 1.1 – Переваги автоматизації

До автоматизації	Після автоматизації
Високий рівень пересортиці товарів; Постійні залишки некондиційних та неліквідних товарів.	Зменшення пересортиці товарів на 70%; Чіткий контроль термінів придатності товарів; Ефективне управління товарними запасами, що дозволяє звести до нуля залишки неліквідних товарів.
Замовлення товарних запасів здійснюється продавці чи керуючі магазинів здійснюють, керуючись інтуїцією або суб'єктивними мотивами, а не реальними потребами бізнесу.	Оперативний аналіз складських залишків та динаміки продажу; Об'єктивний автоматизований розрахунок потреб у товарних запасах; Вчасне поповнення складських запасів
Низька швидкість обслуговування покупців, низька пропускна здатність на касі чи за прилавком;	Збільшення потоку клієнтів на 30%; Зменшення часу обслуговування покупців у два рази; Задоволені покупці.

Продовження таблиці 1.1

Невідомо, які товари дають найбільший прибуток, а продаж яких потрібно стимулювати.	Можна швидко класифікувати товари або групи товарів (ABC/XYZ аналіз) за валовим прибутком та рентабельністю на основі даних торговельної діяльності; Легко планувати акції для стимулювання продажу товарів.
Невідомо, хто з покупців дає найбільшу виручку, валовий дохід від продажу; Відсутня або недієва система лояльності до своїх покупців (бонуси, знижки).	Дуже просто сегментувати своїх покупців (ABC/XYZ аналіз) і ефективно управляти системою лояльності (надавати бонуси, знижки, запроваджувати дисконтні програми);
Всі продажі записують у зошит; Важко контролювати можливі продажі у збиток; Відсутність оперативних аналітичних звітів, рутина та помилки при їх підготовці.	Всю інформацію про продажі роками накопичують у базі даних і на їх основі легко роблять швидкий і всесторонній аналіз; Легко формувати оперативні звіти щодо обсягу, валового прибутку, рентабельності продажів у різноманітних розрізах та з допомогою різної аналітики.

Отже, автоматизація несе в собі багато позитивних якостей, тому в наш час на багатох підприємствах використовуються наведені методи поліпшення роботи. На сучасному етапі, при планування діяльності підприємства потрібно враховувати автоматизацію. Для кожного підприємства вона повинна залежити від виду діяльності.

1.2 Сутність автоматизації ПП «Кулішенко Н.А.»

Кожен рік у сфері автоматизації окремих галузей діяльності людини відбуваються значні зміни, які в свою чергу ведуть до змін у свідомості

людей. Відчувши, що використання автоматизованих засобів у багато разів полегшує роботу і при цьому тільки покращує її якість, досить важко відмовитися від придбання і впровадження комп'ютера.

Системи автоматизації торговельної діяльності в останні роки повільно, але вірно займають своє місце і в цій сфері діяльності людей. Автоматизація обліку наявних та відпущених товарів, розрахунків і створення звітів у багато разів підвищують ефективність і якість роботи, значно полегшують працю працівників [2].

З поширенням мережі Інтернет виникли «електронні магазини» торгують самими різними товарами. У порівнянні зі звичайними магазинами вони мають безліч переваг, які сприяють зростанню доходів у цій сфері торгівлі.

Темп сучасного бізнесу не дозволяє відкладати процес реєстрації руху товарів. Більш того, автоматизація у торгівлі важлива не тільки як засіб тотального обліку товару в інтересах власників бізнесу, але, перш за, все в інтересах покупців як засіб покращення швидкості та якості обслуговування, звільнення часу продавця, щоб вільно спілкуватися з покупцем [12].

На сьогоднішній день для автоматизації процесу організації продажу та обліку товарів підприємство використовує переважно засоби табличних редакторів, а у багатьох випадках проведення обіку наявності товарів проводиться вручну.

Процес взаємодії між точками продажу товарів відбувається за допомогою телефону, тому своєчасність оновлення залишків товарів є поганою.

Інтернет - магазин як засіб автоматизації має наступні переваги:

1. Велика аудиторія:

Вже зараз кількість користувачів Інтернету в Україні становить 22 мільйонів чоловік. У порівнянні з «охопленням» звичайного магазинчика, який складається з декількох прилеглих вулиць, це просто величезна цифра. І

до всіх ви можете «достукатися» за допомогою інструментів онлайн-маркетингу - географічні кордони в Мережі відсутні.

2. Доступність 24/7:

Потенциальный покупатель может зайти на сайт интернет-магазина в любое время суток и с любого места. Он доступен в выходные и праздники, а чтобы выбрать нужный товар, не нужно тратить время на поездку в другой конец города, все можно сделать с удобного для человека места.

3. Невеликий бюджет:

Не потрібно брати багатомільйонні кредити, щоб викупити приміщення і зробити в ньому ремонт. Відсутні витрати на комунальні послуги та армію продавців-консультантів. Немає необхідності щомісяця витрачатися на оренду і безліч інших статей.

4. Можливості аналітики:

В онлайні можна відстежувати практично всі і отримані обсяги інформації використовувати для розвитку власного бізнесу. Всі замовлення фіксуються системою, дані про клієнтів зберігаються, і ніщо ніде не губиться.

За допомогою аналітики можна не тільки визначити найбільш ефективні напрямки роботи та оцінити успішність рекламної кампанії, але і в рази знизити вартість залучення нового покупця.

5. Психологічний комфорт:

Багато людей не люблять, коли до них проявляють зайву увагу продавці-консультанти, і вважають за краще робити покупки в інтернет-магазині, де ніхто не відволікає їх і не заважає. Вони можуть вибирати стільки часу, скільки побажають, і не відчувати зайвого психологічного тиску.

6. Функціональність:

Завдяки технічним можливостям сучасних веб-технологій, магазин в Інтернеті пропонує безліч корисних функцій, які покупець може використовувати для пошуку і вибору товарів: розширений пошук,

порівняння характеристик продуктів, фільтри за цінами і іншими показниками.

7. Інформативність:

Можливості інтернет-магазину в цьому плані просто величезні, адже на сторінках сайту ви можете розмістити всю необхідну про товар інформацію, додати фотографії, відео і створити розділ з повноцінними оглядами. Крім того, потенційні покупці дуже цінують відгуки інших людей, особливо якщо вони вже є власниками цікавого їм продукту. Все це позитивно впливає на продажі [14].

Як бачите, переваги у інтернет-магазину як бізнесу досить відчутні, при цьому, якщо у вас вже є звичайна торговельна точка, відкриття на додаток до неї ще й онлайн-магазину здатне вивести бізнес на новий рівень [3].

Виходячи з поставлених задач підприємством і, що механізм віддаленого клієнтського обслуговування та обліку наявності товарів на підприємстві розвинен слабко, було прийняте рішення розробки веб – сайту з можливістю доступу до бази даних для працівників.

1.3 Функції, що потребують автоматизації

Для початку, на етапі проектування потрібно визначити які функції потрібно автоматизувати.

Створений сайт повинен мати відкритий доступ для працівників та керівних органів, натомість для клієнтів функціонал сайту буде обмежений. Усі функції, що будуть реалізовані в сайті повинні бути простими у використанні, навіть у використанні звичайним юзером комп’ютера.

Розроблений прототип програмного додатку повинен виконувати наступні функції:

1. Проведення обліку наявних товарів – сайт повинен забезпечувати безперебійний доступ до наявної інформації про залишок на скалді для співробітників даного підприємства.

2. Внесення нових партій товарів до баз даних – забезпечення працівникам можливості внесення нових товарів до каталогу бази даних, використовуючи виданий логін та пароль, для доступу до даних функцій.

3. Продаж товарів за допомогою мережі інтернет – забезпечення цілодобової можливості клієнтам використання сайту для зручних покупок товарів у мережі інтернет.

4. Можливість реєстрації постійних клієнтів – передбачити для постійних клієнтів спрощенний спосіб покупки товарі, а саме постійно не вводити інформацію про себе та доставку [9].

5. Сортування товарів за категоріями – забезпечити сайт сортуванням товарів по категоріям. Ця функція полегшить користування та пошуком товарів на сайті для клієнтів.

6. Розробка віртуальної корзини – саме ця функція забезпечить клієнтів зручним способом покупки декількох товарів одночасно. Також, ця фунція надає змогу не вводити данні про клієнта та доставку окремо на кожен товар.

7. Фунція обслуговування заказів – реалізація доступа співробітників до поточних заказів, та змоги змінювати її статус і просматрювати заказ для подальшої обробки.

Окрім цього сайт повинен відкриватися навіть на старих версіях браузерів та операційних системах, тому потрібно враховувати технології що дасть доступ для сайту зі старих комп’ютерів.

1.4 Формування вимог до системи

Аналіз вимог полягає в визначенні потреб та умов, які висуваються щодо нового, чи зміненого продукту, враховуючи можливо конфліктні вимоги різних замовників, таких як користувачі чи бенефіціари.

Аналіз вимог є критичним для успішної розробки проекту. Вимоги мають бути задокументованими, вимірними, тестовними, пов'язаними з бізнес-потребами, і описаними з рівнем деталізації достатнім для конструювання системи.

При розробці програмного додатку до нього будуть ставитися вимоги відповідно до моделі FURPS+.

Класифікація вимог до системи FURPS + була розроблена Робертом Грейді з Hewlett-Packard і запропонована в 1992 році. Скорочення FURPS розшифровується так [4]:

1. Functionality, функціональність;
2. Usability, зручність використання;
3. Reliability, надійність;
4. Performance, продуктивність;
5. Supportability, підтримувані;
6. + Необхідно пам'ятати про такі можливі обмеження, як:
   a) обмеження проектування, design;
   b) обмеження розробки, implementation;
   c) обмеження на інтерфейси, interface;
   d) фізичні обмеження, physical.

Якщо застосувати до цієї класифікації популярне поділ вимог на функціональні і нефункціональні, то до них слід віднести всі перераховані вище групи крім першої, тобто URPS +. На рисунку 1.1 наведено функціональний та нефункціональний поділ вимог FURPS+.

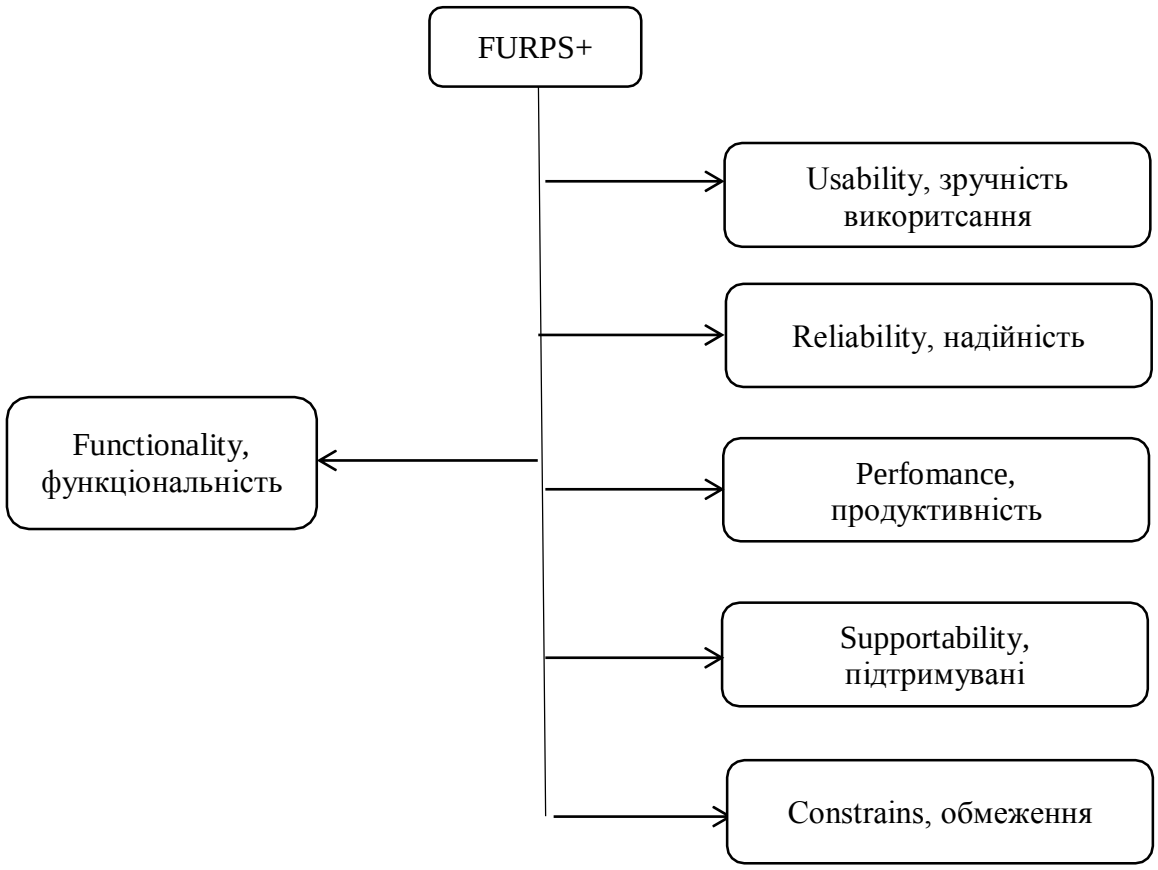


Рисунок 1.1 - Функціональний та нефункціональний поділ вимог FURPS+.

Функціональні вимоги містять основні властивості / функції системи. Однак не всі вони обов'язково будуть ставитися до предметної області системи.

Функціональні вимоги:

1. Журнал роботи, auditing - інструменти відстеження дій користувачів і системи шляхом запису в журнал безпеки конкретних типів подій;
2. Ліцензування, licensing - кошти для відстеження, придбання, установки і контролю над використанням ліцензій;
3. Локалізація, localization - кошти підтримки різних природних мов;
4. Пошта, mail - служби відправки та отримання повідомлень.
5. Допомога, online help - можливість надавати підтримку користувачів в реальному часі;

6. Друк, printing - кошти для друку документів;

7. Звітність, reporting - інструменти створення і отримання звітів;

8. Безпека, security - засоби захисту доступу до певних ресурсів інформації;

9. Управління системою, system management - інструменти, що дозволяють керувати програмами в розподіленої середовищі;

10. Технологічний процес, workflow - підтримка документообігу, включаючи процеси перевірки, візування та затвердження.

Зручність використання.

До зручності використання належать такі види вимог:

1. Естетика і логічність призначеного для користувача інтерфейсу;
2. Захист від людського фактора;
3. Експлуатаційна документація, її склад (посібники, адміністраторів та ін.), галузеві і держ. стандарти оформлення;
4. Кваліфікація користувачів і їх навчання;
5. Довідкова інформація в системі.

Надійність:

Надійність включає такі характеристики системи, як:

1. Збої;
2. Допустима частота / періодичність збоїв;
3. Середній час збоїв і їх серйозність;
4. Можливість відновлення системи після збоїв, в т.ч. можливість попереднього резервного копіювання даних;
5. Передбачуваність поведінки;
6. Час готовності системи до роботи, режим роботи або час доступності системи (наприклад, «Система повинна бути доступна 24 години на добу 7 днів на тиждень»);
7. Точність обчислень.

Продуктивність.

Продуктивність системи складають наступні характеристики:

1. Швидкість роботи, час відгуку системи;
2. Результативність / ефективність;
3. Пропускна здатність, включаючи загальне і допустима кількість одночасно працюючих користувачів, кількість призначених для користувача запитів, число звернень системи до БД і обсяг запитуваних / переданих даних в одиницю часу;
4. Час, необхідний на відновлення - швидкість відновлення;
5. Час, необхідний для запуску і завершення роботи - швидкість запуску і завершення;
6. Споживання ресурсів.

Підтримування системи.

До підтримки відносяться можливості:

1. Тестування;
2. Розширення - нарощування додаткового функціоналу системи;
3. Масштабування - тиражування, наприклад, в філіях / підрозділах організації;
4. Адаптація / пристосування до використання в заданому середовищі, в т.ч. шляхом попередньої настройки;
5. Сумісність;
6. Супроводу, підтримки працездатності: виправлення помилок, оновлення даних, частота архівації та резервного копіювання;
7. Сервісне обслуговування та ремонт;
8. Установка;
9. Локалізації;
10. Портативність;
11. Відповідність міжнародним стандартам.

Вже згадана класифікація + обмеженнявиділяє наступні 4 групи обмежень:

1. Обмеження проектування;

2. Обмеження реалізації, розробки, побудова, написання програмного коду;

3. Вимоги до інтерфейсів - обмеження (наприклад, на формати, протоколи) накладаються необхідністю взаємодії з іншими системами;

4. Фізичні обмеження, що накладаються на технічні (апаратні) засоби.

При розробці системи можуть накладатися і інші обмеження, зокрема юридичні:

1. Міжнародні угоди: одиниці виміру, мови;
2. Авторське право;
3. Договори для отримання ліцензій;
4. Законодавство;
5. Галузеві стандарти.

Отже, було вирішено використовувати вимоги що відносяться до FRUPS+. Саме ця модель зробить проект кращим для використ

2 РОЗРОБКА ПРОЕКТУ АВТОМАТИЗАЦІЇ

2.1 Архітектура та технології вирішення задачі

Інтернет - магазин (електронний магазин, онлайновий магазин) - це програмний комплекс, який дозволяє продавати товари чи послуги через мережу Інтернет та автоматизувати управління бізнес-процесами. Електронні магазини об'єднують елементи прямого маркетингу та традиційної торгівлі. Основними відмінностями Інтернет-магазину від традиційного є інтерактивність, велика кількість інформації та асортименту продукції і персоналізований підхід до кожного відвідувача. Найбільшим недоліком електронних магазинів є те, що не можна торкнутися товару та оцінити його візуально. Проте, цей недолік з успіхом компенсується великою кількістю інформації, яку не зможе надати продавець в традиційному магазині [20].

Для вирішення даної задачі були обрані наступні технології та методики:

1. Язик гіпертекстової розмітки HTML 5:

– HTML — Мова розмітки гіпертекстових документів— стандартна мова розміткивеб-сторінок в Інтернеті. Більшість веб-сторінок створюються за допомогою мови HTML . Документ HTML оброблюється браузером та відтворюється на екрані у звичному для людини вигляді [6];

– HTML є похідною мовою від SGML, успадкувавши від неї визначення типу документа та ідеологію структурної розмітки тексту. Попри те, що HTML — штучна комп'ютерна мова, вона не є мовою програмування. HTML разом із каскадними таблицями стилів та вбудованими скриптами — це три основні технології побудови веб-сторінок.

HTML впроваджує засоби для:

1) створення структурованого документа шляхом позначення структурного складу тексту: заголовки, абзаци, списки, таблиці, цитати та інше;

2) отримання інформації із Всесвітньої мережі через гіперпосилання;

3) створення інтерактивних форм;

4) включення зображень, звуку, відео, та інших об'єктів до тексту [28].

2. Мова CSS3:

Каскадні таблиці стилів — спеціальна мова, що використовується для опису зовнішнього вигляду сторінок, написаних мовами розмітки даних.

Найчастіше CSS використовують для візуальної презентації сторінок, написаних HTML та XHTML, але формат CSS може застосовуватися до інших видів XML-документів.

CSS має різні рівні та профілі. Наступний рівень CSS створюється на основі попередніх, додаючи нову функціональність або розширюючи вже наявні функції. Рівні позначаються як CSS1, CSS2 та CSS3. Профілі — сукупність правил CSS одного або більше рівнів, створені для окремих типів пристроїв або інтерфейсів.

CSS прийшла на заміну табличній верстці веб-сторінок. Головна перевага блочної верстки — розділення змісту сторінки (даних) та їхньої візуальної презентації [7].

CSS використовується авторами та відвідувачами веб-сторінок, щоб визначити кольори, шрифти, верстку та інші аспекти вигляду сторінки. Одна з головних переваг — можливість розділити зміст сторінки (або контент, наповнення, зазвичай HTML, XML або подібна мова розмітки) від вигляду документу (що описується в CSS).

Таке розділення може покращити сприйняття та доступність контенту, забезпечити більшу гнучкість та контроль за відображенням контенту в різних умовах, зробити контент більш структурованим та простим. CSS

також дозволяє адаптувати контент до різних умов відображення (на екрані монітора, мобільного пристрою , у роздрукованому вигляді, на екрані телевізора, пристроях з підтримкою шрифту Брайля або голосових браузерах та ін.).

Один і той самий HTML або XML документ може бути відображений по-різному залежно від використаного CSS. Стилі для відображення сторінки можуть бути:

- зовнішні таблиці стилів, найчастіше окремий файл або файли css;
- внутрішні таблиці стилів, включені як частина документу або блоку;
- стилі для окремого елементу;
- стилі користувача.

Локальний css-файл, вказаний користувачем для використання на сторінках і вказаний в налаштуваннях браузера.

Стандартний стиль переглядача, наприклад стандартні стилі для елементів, визначені браузером, використовуються коли немає інформації про стиль елемента або вона неповна.

Стандарт CSS визначає порядок та діапазон застосування стилів, тобто, в якій послідовності і для яких елементів застосовуються стилі. Таким чином, використовується принцип каскадності, коли для елементів вказується лише та інформація про стилі, що змінилася або не визначена загальнішими стилями.

Переваги CSS:

1. Інформація про стиль для усього сайту або його частин може міститися в одному css-файлі, що дозволяє швидко робити зміни в дизайні та презентації сторінок;
2. Різна інформація про стилі для різних типів користувачів: наприклад великий розмір шрифту для користувачів з послабленим зором, стилі для виводу сторінки на принтер, стиль для мобільних пристроїв;

3. Сторінки зменшуються в об'ємі та стають більш структурованими, оскільки інформація про стилі відділена від тексту та має певні правила застосування і сторінка побудована з урахуванням їх;

4. Прискорення завантаження сторінок і зменшення обсягів інформації, що передається, навантаження на сервер та канал передачі. Досягається за рахунок того, що сучасні браузери здатні кешувати (запам'ятовувати) інформацію про стилі і використовувати для всіх сторінок, а не завантажувати для кожної [21].

CSS має порівняно простий синтаксис і використовує небагато англійських слів для найменування різних складових стилю.

Стилі складаються зі списку правил. Кожне правило має один або більше селектор та блок визначення.

Властивості в списку оформлюються у вигляді назва властивості, двокрапка (:), значення, крапка з комою (;).

Язик програмування PHP.

PHP — 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.

PHP інтерпретується веб-сервером у HTML-код, який передається на сторону клієнта.

PHP — мова, у код якої можна вбудовувати безпосередньо html-код сторінок, які, у свою чергу, коректно оброблюватимуться PHP-інтерпретатором. Обробник PHP просто починає виконувати код після відкриваючого тегу (<?php) і продовжує виконання до того моменту, поки не зустріне закриваючий тег (?>).

Також у PHP вбудовані бібліотеки для роботи багатьох баз даних, а саме:MySQL, PostgreSQL, mSQL, Oracle, dbm, Hyperware, Informix, InterBase, Sybase. Завдяки стандарту відкритого інтерфейсу зв'язку з базами даних можна підключатися до всіх баз даних, до яких існує драйвер.

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 мовами C або Pascal. Це помітно знижує початкові зусилля при вивченні PHP. PHP — мова, що поєднує переваги Perl та 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 найпопулярнішою мовою для створення веб-застосунків(скриптів).

Ухвалення стратегії Open Source і безкоштовне розповсюдження початкових текстів PHP надало неоціненну послугу користувачам. Окрім цього, користувачі PHP в усьому світі є свого роду колективною службою підтримки, і в популярних електронних конференціях можна знайти відповіді, навіть на найскладніші питання.

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 обробляються швидше за аналогічні їм програми [22].

SQL (англ. Structured query language — мова структурованих запитів) — декларативна мова програмування для взаємодії користувача з базами даних, що застосовується для формування запитів, оновлення і керування реляційними БД, створення схеми бази даних та її модифікації, системи контролю за доступом до бази даних. Сама по собі SQL не є ані системою керування базами даних, ані окремим програмним продуктом. На відміну від дійсних мов програмування, SQL може формувати інтерактивні запити або, бувши вбудованою в прикладні програми, виступати як інструкції для керування даними. Окрім цього, стандарт SQL містить функції для визначення зміни, перевірки та захисту даних.

SQL — це діалогова мова програмування для здійснення запиту і внесення змін до бази даних, а також керування базами даних. Багато баз даних підтримує SQL з розширеннями до стандартної мови. Ядро SQL формує командна мова, яка дозволяє здійснювати пошук, вставку, оновлення і вилучення даних за допомогою використання системи керування і адміністративних функцій. SQL також включає CLI (Call Level Interface) для доступу і керування базами даних дистанційно [5].

Перша версія SQL була розроблена на початку 1970-х років у IBM. Ця версія мала назву SEQUEL і була призначена для обробки та пошуку даних, що містилися в реляційній базі даних IBM, System R. Мова SQL надалі була стандартизована Американськими Держстандартами в 1986. На початку SQL була запланована як мова запитів і управління даними, а пізніші модифікації SQL створені продавцями системи управління базами даних, які додали процедурні конструкції, control-of-flow команд і розширення мов. З випуском стандарту SQL:1999 такі розширення були формально запозичені як частина мови SQL через Persistent Stored Modules (SQL/PSM).

Переваги SQL:

1. Незалежність від конкретної СУБД - не зважаючи на наявність діалектів і відмінностей в синтаксисі, більшість текстів SQL-запитів, що містять, DDL і DML, можуть бути досить легко перенесені з однієї СУБД в іншу. Існують системи, розробники яких спочатку орієнтувалися на застосування щонайменше кількох СУБД. Природно, що при застосуванні деяких специфічних для реалізації можливостей, такого рівня перенесення дуже важко досягти;

2. Наявність стандартів - наявність стандартів і наборів тестів для виявлення сумісності та відповідності конкретній реалізації SQL загальноприйнятому стандарту тільки сприяє «стабілізації» мови. Щоправда, слід звернути увагу на той факт, що сам по собі стандарт місцями занадто формалізований і має завеликі розміри, наприклад, Core-частина стандарту SQL:2003 містить понад 1300 сторінок тексту;

3. Декларативність - за допомогою SQL програміст описує лише дані, які потрібно витягнути або модифікувати. Те, яким чином це зробити, вирішує СУБД безпосередньо при обробці SQL-запиту. Не слід вважати, що це повністю універсальний принцип — програміст описує набір даних для вибірки або модифікації, проте йому при цьому корисно уявляти, як СУБД інтерпретуватиме текст його запиту. Такі моменти стають особливо критичними при роботі з великими базами даних та зі складними запитами — чим складніше сконструйований запит, тим більше варіантів написання він припускає.

Недоліки SQL :

1. Невідповідність реляційної моделі даних - творець реляційної моделі даних Едгар Кодд, Крістофер Дейт та їхні прихильники вказують на те, що SQL не є істинно реляційною мовою. Зокрема, вони привертають увагу до таких проблем SQL:

- рядки, що повторюються;
- невизначені значення (null);
- явна вказівка порядку стовпчиків зліва направо;
- стовпчики без імені та імена стовпчиків, що дублюються;
- відсутність підтримки властивості «=»;
- використання вказівників;
- значна надлишковість.

В опублікованому Крістофером Дейтом і Г'ю Дарвеном Третьому Маніфесті, вони декларують принципи СУБД наступного покоління та пропонують мову Tutorial D, яка є достовірно реляційною [15].

2. Складність:

Хоча мову SQL було початково заплановано як засіб роботи кінцевого користувача, врешті-решт вона стала настільки складною, що перетворилася на інструмент програміста.

3. Відступи від стандартів:

Не зважаючи на наявність міжнародного стандарту ANSI SQL-92, багато компаній, що займаються розробкою СУБД (наприклад, Oracle, Sybase, Microsoft, MySQL), вносять зміни до мови SQL, вживаної в розроблених ними СУБД. Цим вони створюють передумови відступу від стандарту. Завдяки такій діяльності для кожної конкретної СУБД з'являються специфічні діалекти мови SQL[33].

4. Складність роботи з ієрархічними структурами.

Раніше SQL не пропонувала стандартного способу маніпуляції деревовидними структурами. Деякі постачальники СУБД запропонували свої рішення. Для прикладу, Oracle використовує вираз CONNECT BY. В наш час, як стандарт прийнята рекурсивна конструкція WITH.

Оскільки SQL не є мовою програмування (тобто не надає засобів для автоматизації операцій з даними), нововведення різних виробників стосувалися в першу чергу процедурних розширень. Це збережені процедури і процедурні мови - «надбудови». Практично в кожній СУБД застосовується своя процедурна мова. Подібні мови для найпопулярніших СУБД приведені в наступній таблиці [30].

Таблиця 2.1 – Найпопулярніші СУБД

СУБД	Коротка назва	Розшифрування
Borland InterBase/Firebird	PSQL	Procedural SQL
IBM DB2	SQL PL	SQL Procedural Language (розширює SQL/PSM)
Microsoft SQL Server/b Sybase ASE	Transact-SQL	Transact-SQL
MySQL	SQL/PSM	SQL/Persistent Stored Module
Oracle	PL/SQL	Procedural Language/SQL
PostgreSQL	PL/pgSQL	Procedural Language/PostgreSQL Structured Query Language

Визначивши недоліки та переваги розширень баз даних та SQL, для розробки проекту було обрано СУБД MySQL.

Bootstrap - це безкоштовний набір інструментів з відкритим кодом, призначений для створення веб-сайтів та веб-додатків, який містить шаблони CSS та HTML для типографіки, форм, кнопок, навігації та інших компонентів інтерфейсу, а також додаткові розширення JavaScript. Він спрощує розробку динамічних веб-сайтів і веб-додатків [32].

Bootstrap — це клієнтський фреймворк, тобто інтерфейс для користувача, на відміну від коду серверної сторони, який знаходиться на сервері.

Bootstrap має модульну структуру і складається переважно з наборів таблиць стилів LESS, які реалізують різні компоненти цього набору інструментів. Розробники можуть самостійно налаштовувати файли Bootstrap, обираючи компоненти для свого проекту.

Основні інструменти Bootstrap:

1. Сітки (grid) — наперед задані, готові до використання колонки;
2. Шаблони (template) — фіксовані чи адаптивні шаблони сторінок;
3. Типографіка (typography) — опис та визначення класів для шрифтів, таких як шрифти для коду, цитат тощо;
4. Мультимедіа (media) — засоби управління зображеннями та відео;
5. Таблиці (table) — засоби оформлення таблиць, які зокрема забезпечують сортування;
6. Форми (form) — класи для оформлення як форм, так і деяких подій;
7. Навігація (nav, navbar) — класи для оформлення вкладок, сторінок, меню і панелей навігації;
8. Сповіщення (alert) — класи для оформлення діалогових вікон, підказок і спливаючих вікон;
9. Іконочний шрифт (icon font) — набір іконок у вигляді шрифту, складається майже з 500 компонентів [24].

Окрім стилів, фреймворк містить також функціональні компоненти, які побудовані на js з використанням jQuery і містять такі плагіни:

1. Transitions — плавні зміни, плагін використовується для налаштування останніх компонентів фреймворку;
2. Modal — модальні вікна, як спливні, так і вбудовані в сторінку;
3. Dropdown — випадні списки, побудовані без тегу select ;
4. Scrollspy — плагін, що автоматично міняє активний пункт в меню в залежності від позиції прокрутки сторінки;
5. Tab — вкладки, використовується зазвичай для стилізованої навігації;
6. Tooltip — спливні підказки, текстові елементи, що з'являються поряд із вказаним об'єктом після наведення курсору;
7. Popover — аналог спливних підказок, але з більшими можливостями. У підказку можна додавати заголовок, до того ж блок з'являється після кліку на об'єкті;
8. Alert — інформаційні повідомлення, які створюються класом .alert , але з можливістю закриття;
9. Button — плагін для керуваннями станами кнопок. Завдяки методам плагіну можна міняти стан і тип кнопки, а також створювати елементи, що поводяться як checkbox або radio button , але при цьому є звичайними блочними елементами;
10. Collapse — згортання блочних елементів;
11. Carousel — мультимедійна галерея зображень;
12. Affix — плагін, що «приліплює» меню до одного з країв екрану при прокрутці сторінки.

JQuery. Популярна JavaScript-бібліотека з відкритим сирцевим кодом. Вона була представлена у січні 2006 року у BarCamp NYC Джоном Ресіґом. Згідно з дослідженнями організації W3Techs, JQuery використовується понад половиною від мільйона найвідвідуваніших сайтів. jQuery є

найпопулярнішою бібліотекою JavaScript, яка посилено використовується на сьогоднішній день.

jQuery є вільним програмним забезпеченням під ліцензією MIT. 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 [11].

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, ефекти, і Ajax функції. Вона може бути додана до web-сторінки посиланням на локальну копію, або на одну з копій доступних на публічному сервері.

2.2 Технічне забезпечення автоматизованого програмного рішення

Для того, щоб автоматизована система ефективно працювала потрібно встановити відповідне програмне та апаратне забезпечення. До складу програмного забезпечення входять спеціальні програмні продукти, а також документація з експлуатації.

До загальносистемного програмного забезпечення відносять програми, які орієнтовані на користувачів і призначених для вирішення типових завдань обробки інформації. Вони направлені на розширення функціональних можливостей комп'ютерів, контролю та управління процесом обробки даних.

До апаратного забезпечення належать пристрої, що утворить конфігурацію комп'ютера. Розрізняють внутрішні і зовнішні пристрої.

Узгодження між окремими вузлами і блоками виконується за допомогою апаратно-логічних пристроїв, званих апаратними інтерфейсами.

На підприємстві використовується 4 комп'ютера, з подальшим розвитком планується збільшення кількості пристроїв.

В якості програмного забезпечення для комп'ютера працівників чи користувача рекомендується Windows 7 чи новіше, MacOs 10.9 чи новіше.

Мінімальна конфігурація компютара на базі операційної системи Windows 7 чи новіше для використання автоматизованої системи таблиця 2.2

Таблиця 2.2 – Мінімальні характеристики робочої станції з Windows

Назва складової	Характеристика
Екран	Full HD
Процесор	Intel Core 2 Duo 2 ГГц
Оперативна пам'ять	2 Гб DDR3
Графічний адаптер	Вбудований
Жорсткий диск	128 Гб
Мережа	Підключення до мережі зі швидкістю 1мб/с

Мінімальна конфігурація компютара на базі операційної системи MacOS 10.12 чи новіше або Linux - дистрибутивів для використання автоматизованої системи таблиця 2.3.

Таблиця 2.3 – Мінімальні характеристики робочої станції з MacOS або Linux – дестрибутивів.

Назва складової	Характеристика
Екран	Full HD
Процесор	Intel Core 2 Duo 1,86 ГГц
Оперативна пам’ять	2 Гб DD3
Графічний адаптер	Вбудований
Жорсткий диск	128 Гб
Мережа	Підключення до мережі зі швидкістю 1мб/с

Представлене програмне та апаратне забезпечення надає можливість коректної та ефективної роботи з автоматизованою системою і дозволить виконувати поставлені перед системою завдання.

2.3 Програмне забезпечення автоматизованого рішення

Sublime Text це багатоплатформовий пропріетарний текстовий редактор. Підтримує плагіни на мові програмування Python.

Розробник дозволяє безкоштовно і без обмежень ознайомитися з продуктом, проте програма повідомляє про необхідність придбання ліцензії.

Sublime Text підтримує велику кількість мов програмування і має можливість підсвічування синтаксису для C, C ++, C #, CSS, D, Dylan, Erlang, HTML, Groovy, Haskell, Java, JavaScript, LaTeX, Lisp, Lua, Markdown, MATLAB, OCaml, Perl , PHP, Python, R, Ruby, SQL, TCL і XML [18].

На додаток до тих мов програмування, які включені за замовчуванням, користувачі мають можливість завантажувати плагіни для підтримки інших мов [25].

Sublime Text може бути оснащений менеджером пакетів, який дозволяє користувачеві знаходити, встановлювати, оновлювати і видаляти пакети без перезавантаження програми. Менеджер підтримує встановлені пакети в актуальному стані, завантажуючи нові версії з репозиторіїв. Крім того, він надає команди для активації і деактивації встановлених пакетів [34].

Редактор містить різні візуальні теми, з можливістю завантаження додаткових.

Користувачі бачать весь свій код в правій частині екрану у вигляді міні-карти, при кліці на яку можна здійснювати навігацію.

Є кілька режимів екрану. Один з них включає від 1 до 4 панелей, за допомогою яких можна показувати до чотирьох файлів одночасно. Повноцінний (free modes) режим показує тільки один файл без будь-яких додаткових навколо нього меню.

Виділення стовпців цілком або розстановка кілька покажчиків по тексту, що робить можливим миттєву правку. Покажчики поводяться, ніби кожен з них - єдина в тексті. Команди типу: переміщення на знак, переміщення на рядок, вибірка тексту, переміщення на слово або його частини (CamelCase, розділений дефісом або підкресленням), перехід на початок / кінець рядка і т. Д., Впливає на всі покажчики незалежно і відразу, дозволяючи правити складноструктурованих текст швидко, без використання макрокоманд або регулярних виразів.

Коли користувач набирає код, Sublime Text, в залежності від використовуваної мови, буде пропонувати різні варіанти для завершення запису. Редактор також автоматично завершує створені користувачем змінні.

Темний фон Sublime Text призначений для збільшення контрастності тексту. Основні елементи синтаксису виділені різними кольорами, які краще поєднуються з темним тлом, ніж зі світлим.

Sublime Text дозволяє користувачеві збирати програми і запускати їх без необхідності перемикатися на командний рядок. Користувач також може

налаштувати свою систему збирання та включити автоматичну збірку програми кожного разу при збереженні коду.

Збереження фрагментів часто використовуваного коду, ключові слова для їх запуску.

Навігаційний інструмент, який дозволяє користувачам переміщатися між файлами, а також всередині них, за допомогою нечіткого пошуку.

Додатково реалізована функція автозбереження, яка допомагає користувачам не втратити виконану роботу.

Настроюються комбінації клавіш і інструмент навігації дозволяють призначати свої комбінації клавіш для меню і панелей інструментів (тільки для першої версії, у другій і третій - Command Palette).

Можливість пошуку під час набору використовується для пошуку в документі.

Функція перевірки синтаксису працює подібним же чином, перевіряючи коректність прямо під час введення.

Є можливість автоматизації за допомогою макросів і повтору останніх дій.

Команди редагування, включаючи редагування відступів, переформатування параграфів і об'єднання рядків.

Програмне забезпечення JetBrains RubyMine є повнофункціональну середу розробки для створення додатків на базі популярних технологій Ruby і Ruby on Rails. Продукт має можливості розробки web-додатків середовища IntelliJ IDEA і включає нові унікальні інструменти створення web-проектів на базі платформ Ruby і Rails.

Середа RubyMine надає ефективні інструменти розробки, відповідні мовними особливостями платформи Ruby і забезпечують облік динамічного характеру цієї мови і прийнятих в ньому стандартних методів роботи. Середа RubyMine дозволяє виконувати всебічний аналіз коду проекту, пропонуючи розробникам зведений список коду і унікальні функції автоматичного завершення мовних конструкцій. Продукт RubyMine пропонує широкий

набір алгоритмів рефакторинга з урахуванням особливостей платформи Rails, що значно полегшує редагування коду і прискорює процес розробки продукту [26].

Середовище включає можливість зручного перетворення коду HTML, CSS і JavaScript.

Середовище розробки RubyMine підтримує інтеграцію з системами управління версіями коду, включаючи платформи Subversion, Git, Perforce і CVS. Графічний відладчик для Ruby і Rails дозволяє встановлювати точки зупинки в коді Ruby і файлах відображення Rails RHTML. Підтримка платформ тестування RSpec і Test :: Unit допомагає організувати тестування модулів за допомогою графічного середовища для проведення тестів. Наявність відкритого програмного інтерфейсу API дозволяє створювати власні надбудови на мові Ruby або Java.

Програмне забезпечення JetBrains WebStorm являє собою інструмент для розробки web-сайтів і редагування HTML, CSS і JavaScript коду.

Рішення забезпечує швидку навігацію по файлах і генерує повідомлення про виникаючі проблеми в коді в режимі реального часу. JetBrains WebStorm дозволяє додавати розмітку HTML-документів або елементів SQL безпосередньо в JavaScript. JetBrains WebStorm здійснює розгортання і синхронізацію проектів через протокол FTP.

Використовуючи можливості коду HTML / XHTML і XML, WebStorm забезпечує автоматичне завершення стилів, посилань, атрибутів і інших елементів коду.

При роботі з CSS здійснюється завершення коду класів, HTML-номерів, ключових слів і т. Д. WebStorm пропонує автоматичне рішення таких проблем, як вибір формату, властивостей, класів, посилань на файли і інших атрибутів CSS. Рішення дозволяє використовувати потужність інструменту Zen coding для верстки HTML, відображає дії тега на web-сторінці.

Продукт WebStorm здійснює завершення коду JavaScript для ключових слів, лейблів, змінних, параметрів і функцій DOM і підтримує специфічні особливості популярних браузерів. Реалізовані в рішенні функції рефакторінга JavaScript дозволяють перетворювати структуру коду і файлів і .js [27].

WebStorm забезпечує налагодження коду JavaScript і надає широкий діапазон можливостей: знаходження точки зупину в HTML і JavaScript, настройка параметрів точки зупину, тестування синтаксису коду в режимі реального часу і т. Д. Продукт підтримує платформи JQuery, YUI, Prototype, DoJo, MooTools, Qooxdoo і Bindows.

WebStorm передбачає інтегровану перевірку тексту на теги, послідовність коду, помилки в написанні і т. Д. WebStorm дозволяє редагувати файли і автоматично синхронізувати їх на вимогу при віддаленій роботі або зберіганні.

Продукт підтримує функцію контролю версій і попередніх варіантів коду і фіксує всі вироблені дії та зміни. Завдяки створенню історії в WebStorm можна відновлювати кодові вирази, блоки і навіть цілі файли.

Система управління базами даних MySQL - розробка шведської компанії MySQL AB. СУБД MySQL є програмним забезпеченням з відкритим вихідним кодом, поширюваним за ліцензією GNU (GPL) і комерційної ліцензії для ситуацій, які не підпадають під дію ліцензії GPL. MySQL підтримує реляційну модель даних, т. Е. Є реляційною СУБД.

Починаючи з версії 5.0, СУБД MySQL практично повністю задовольняє стандарту структурованого мови запитів SQL і, отже, сумісна з іншими базами даних.

Розглянемо основні переваги СУБД MySQL:

1. Висока якість - MySQL характеризується стійкою роботою;

2. Поряд з Oracle, MySQL вважається однією з найшвидших СУБД в світі;

3. Відкритий код доступний для перегляду і модернізації, що дозволяє постійно покращувати програмний продукт;

4. СУБД MySQL, розроблена з використанням мов C / C ++, протестована на багатьох платформах, серед яких Windows, Linux, FreeBSD, Mac OS X, OS / 2, Solaris і ін;

5. MySQL підтримує API (програмний інтерфейс програми) для C, C ++, Eiffel, Java, Perl, PHP, Python, Ruby і Tcl. MySQL можна успішно застосовувати як для побудови Web-сторінок з використанням Perl, PHP і Java, так і для роботи прикладної програми, створеної з використанням Delphi, Builder C ++ або платформи .NET;

6. СУБД MySQL надає широкий вибір типів таблиць, в тому числі і сторонніх розробників, що дозволяє реалізувати оптимальну для розв'язуваної задачі продуктивність і функціональність;

7. Локалізація в MySQL виконана коректно. У користувача, як правило, не виникає проблем при обробці російського вмісту БД.

На офіційному сайті MySQL можна знайти доступні для скачування версії цього продукту.

Ще в версії MySQL 4.1 з'явилися такі важливі нововведення, як повна підтримка вкладених запитів і підтримка транзакцій. А у версії MySQL 5.0 стали доступними такі важливі механізми:

1) процедури, що і функції, які об'єднують в собі цілі послідовності запитів;

2) тригери - збережені процедури, прив'язані до події зміни таблиці;

3) уявлення - вибірки даних, які можна уявити як повноцінні реально існуючі таблиці бази даних;

4) курсори, що дозволяють циклі переглянути кожен рядок результуючої таблиці запитів;

5) інформаційна схема - стерпний набір уявлень системної таблиці, в якій зберігається різноманітна внутрішня інформація.

6) обробники помилок;

7) безліч нових функцій.

У стандартному дистрибутиві MySQL поставляються клієнтські програми , які взаємодіють з MySQL-сервером:

1) mysql - консольний клієнт для доступу до MySQL-сервера, що дозволяє виконувати SQL-запити;

2) mysqladmin - утиліта для виконання адміністративних функцій, таких як створення або видалення баз даних , отримання інформації про процеси, стан сервера і т. п.;

3) mysqldump - утиліта для виведення вмісту бази даних MySQL у вигляді текстового файлу з SQL-операторами;

4) mysqlimport - виконує перенесення інформації з текстового файлу в таблиці баз даних;

5) mysqlshow - щоб отримати інформацію про існуючі базах даних, таблицях, полях і індексах.

До утиліт, які можуть функціонувати без підключення до сервера MySQL, відносяться:

1) myisampack - стискає таблиці типу MylSAM, зменшуючи їх в розмірі і роблячи доступними тільки для читання;

2) mysqlcheck - утиліта, яка використовується для опису, перевірки, оптимізації та відновлення таблиць;

3) mysqlbinlog - дана утиліта використовується для читання вмісту журналу двійковій реєстрації при відновленні даних в нештатних ситуаціях

4) perror - утиліта, яка виводить розшифровку кодів системних помилок і помилок MySQL.

Денвер - набір дистрибутивів (локальний сервер WAMP) і програмна оболонка, призначені для створення і налагодження сайтів (веб-додатків, іншого динамічного вмісту інтернет-сторінок) на локальному ПК (без необхідності підключення до мережі Інтернет) під керуванням ОС Windows.

Пакет поширюється як freeware . Відразу після установки доступний повністю працюючий веб-сервер Apache, що працює на локальному

комп'ютері, на якому може працювати необмежену кількість сайтів, що дуже ефективно для розробки і налагодження сценаріїв PHP без завантаження його файлів на віддалений сервер. Для запуску практично всіх утиліт «Денвера» використовується додаток Run в підкаталозі / denwer кореневого каталогу установки «Денвера». При запуску створюється віртуальний диск (за замовчуванням Z :), де зберігаються всі файли проектів.

Третя версія підтримує роботу зі знімного флеш-накопичувача.

Особливістю, що відрізняє Denwer від інших WAMP-дистрибутивів , є автоматична правка системного файлу hosts, що є локальним аналогом DNS-сервера, що дозволяє звертатися до локальних сайтів, які працюють під управлінням Денвера, по іменах, що збігається з ім'ям папки, розташованої в каталозі home.

Базовий пакет:

- веб-сервер Apache з підтримкою SSI, SSL, mod_rewrite, mod_php;
- інтерпретатор PHP з підтримкою GD, MySQL, SQLite;
- СУБД MySQL з підтримкою транзакцій (mysqld-max);
- система управління віртуальними хостами, заснована на шаблонах;
- система управління запуском і завершенням;
- панель phpMyAdmin для адміністрування СУБД;
- ядро інтерпретатора Perl без стандартних бібліотек (поставляються окремо);
- емулятор sendmail і сервера SMTP з підтримкою роботи спільно з PHP, Perl, Parser і ін;
- установник.

Пакети розширень:

- повна версія ActivePerl 5.8;
- інтерпретатор PHP версії 5 з повним набором модулів;
- інтерпретатор Python;
- СУБД MS SQL, PostgreSQL або InterBase / FireBird версій 1 і 2;
- інтерпретатор PHP версії 3 або 4;

– інтерпретатор Parser;
– виправлення в CONFIGURATION.

Саме ці программи доможуть у реалізації проекту. Вони забезпечать високу продуктивність у створенні коду та впровадження проекту на підприємстві.

2.4 Правове забезпечення автоматизованого рішення

Правила охорони праці під час експлуатації електроннообчислювальних машин поширюються на всі підприємства, установи, організації, юридичні особи, незалежно від форми власності, відомчої належності, видів діяльності та на фізичних осіб, які здійснюють розробку, виробництво та застосування електронно-обчислювальних машин і персональних комп'ютерів (далі – ЕОМ), у тому числі й на тих, які мають робочі місця, обладнані ЕОМ, або виконують обслуговування, ремонт та налагодження ЕОМ.

Вимоги безпеки під час експлуатації ЕОМ.

Користувачі ЕОМ повинні слідкувати за тим, щоб відеотермінали, ЕОМ, периферійні пристрої ЕОМ та устаткування для обслуговування, ремонту та налагодження ЕОМ були справними і випробуваними відповідно до чинних нормативних документів. Щоденно перед початком роботи необхідно проводити очищення екрана відеотермінала від пилу та інших забруднень. Після закінчення роботи відеотермінал та персональна ЕОМ повинні бути відключені від електричної мережі. У разі виникнення аварійної ситуації необхідно негайно відключити відеотермінал та ЕОМ від електричної мережі. При використанні з ЕОМ та відеотерміналами лазерних принтерів потрібно дотримуватись вимог Санітарних норм та правил устрою

та експлуатації лазерів № 5804–91, затверджених Міністерством охорони здоров'я в 1991 р.

Вимоги електробезпеки.

Під час проектування систем електропостачання, монтажу силового електрообладнання та електричного освітлення будівель та приміщень для ЕОМ необхідно дотримуватись вимог ПВЕ, ПТЕ, ПБЕ, СН 357–77 «Инструкция по проектированию силового осветительного оборудования промышленных предприятий», затверджених Держбудом, ГОСТ 12.1.006, ГОСТ 12.1.030 «ССБТ. Электробезопасность. Защитное заземление, зануление», ГОСТ 12.1.019 «ССБТ. Электробезопасность. Общие требования и номенклатура видов защиты», ГОСТ 12.1.045, ВСН 59-88 Держкомархитектури «Электрооборудование жилых и общественных зданий. Нормы проектирования», Правил пожежної безпеки в Україні, цих Правил, а також розділів СНиП, що стосуються штучного освітлення і електротехнічних пристроїв, та вимог нормативно–технічної і експлуатаційної документації заводу-виробника ЕОМ. У приміщенні, де одночасно експлуатується або обслуговується більше п'яти персональних ЕОМ, на помітному та доступному місці встановлюється аварійний резервний вимикач, який може повністю вимкнути електричне живлення приміщення, крім освітлення. ЕОМ, периферійні пристрої ЕОМ та устаткування для обслуговування ремонту та налагодження ЕОМ повинні підключатися до електромережі тільки з допомогою справних штепсельних з'єднань і електророзеток заводського виготовлення. Неприпустимим є підключення ЕОМ, периферійних пристроїв ЕОМ та устаткування для обслуговування, ремонту та налагодження ЕОМ до звичайної двопровідної електромережі, в тому числі – з використанням перехідних пристроїв. Електромережу штепсельних розеток для живлення персональних ЕОМ, периферійних пристроїв ЕОМ та устаткування для обслуговування, ремонту та налагодження ЕОМ при розташуванні їх уздовж стін приміщення прокладають по підлозі поряд зі стінами приміщення, як правило, в

металевих трубах і гнучких металевих рукавах з відводами відповідно до затвердженого плану розміщення обладнання та технічних характеристик обладнання [16].

Вимоги до обладнання.

Відеотермінали, ЕОМ, ПЕОМ, спеціальні периферійні пристрої ЕОМ та устаткування для обслуговування, ремонту та налагодження ЕОМ повинні відповідати вимогам чинних в Україні стандартів, нормативних актів з охорони праці та цих Правил. Відеотермінали, ЕОМ, ПЕОМ,v спеціальні периферійні пристрої ЕОМ закордонного виробництва додатково повинні відповідати вимогам національних стандартів держав–виробників і мати відповідну позначку на корпусі, в паспорті або іншій експлуатаційній документації [10].

Вимоги до організації робочого місця користувача ЕОМ.

Організація робочого місця користувача відеотерміналу та ЕОМ повинна забезпечувати відповідність усіх елементів робочого місця та їх розташування ергономічним вимогам ГОСТ 12.2.032 «ССБТ. Рабочее место при выполнении работ сидя. Общие эргономические требования»; характеру та особливостям трудової діяльності [10]. Площа, виділена для одного робочого місця з відеотерміналом або персональною ЕОМ, повинна складати не менше 6 м 2 , а обсяг – не менше 20 м 3 .

Робочі місця з відеотерміналами відносно світлових прорізів повинні розміщуватися так, щоб природне світло падало збоку, переважно зліва.

При розміщенні робочих місць з відеотерміналами та персональними ЕОМ необхідно дотримуватись таких вимог: робочі місця з відеотерміналами та персональними ЕОМ розміщуються на відстані не менше 1 м від стін зі світловими прорізами;відстань між бічними поверхнями відеотерміналів має бути не меншою за 1,2 м;відстань між тильною поверхнею одного відео термінала та екраном іншого не повинна бути меншою 2,5 м; прохід між рядами робочих місць має бути не меншим 1 м.

Висота робочої поверхні столу для відеотермінала має бути в межах 680–800 мм, а ширина – забезпечувати можливість виконання операцій в зоні досяжності моторного поля. Рекомендовані розміри столу: висота –725 мм, ширина –600–1400 мм, глибина –800–1000 мм. Для зниження статичного напруження м’язів рук необхідно застосовувати стаціонарні або знімні підлокітники довжиною не менше 250 мм, шириною – 50 – 70 мм, що регулюються по висоті над сидінням у межах230 ± ЗО мм та по відстані між підлокітниками в межах 350–500 мм. Екран відеотермінала та клавіатура мають розташовуватися на оптимальній відстані від очей користувача, але не ближче 600 мм, з урахуванням розміру алфавітно–цифрових знаків та символів.

3 РЕАЛІЗАЦІЯ ПРОТОТИПУ АВТОМАТИЗАЦІЇ МЕХАНІЗМУ ПРОДАЖУ ТА ОБЛІКУ ТОВАРІВ НА ПІРПРИЄМСТВІ

3.1 Інформаційне забезпечення автоматизації

Сучасні інформаційних технологій висувають нові можливості для підвищення рівня продажу товару через мережу Інтернет. Наповнення вебсайту інтернет-магазину контентом є актуальним питанням на сьогоднішній день, оскільки багато людей кожного дня, не виходячи з дому, купують різні товари в онлайн магазинах. В світі, а зокрема в Україні швидкими та великими темпами росте кількість користувачів мережі Інтернет в наслідок чого, збільшується кількість «електронних» покупців.

Створення вдалого сайту інтернет-магазину це ефективний спосіб для реалізації продукції в електронній торгівлі. Оскільки інтернет- магазини суттєво зменшують витрати не тільки для виробника, який заощаджує на утриманні звичайного магазина, а й для покупця який має можливість придбати товар в будь-який час та в будь-якому місті не виходячи з дому. Веб-сайт інтернет-магазину – це сайт, який здійснює продаж товарів, вибір яких обирається користувачем з каталогу. Перше, що привертає до себе увагу на сайті - це контент, дані (наприклад зображення, відео, текст і т.д) про інтернет-магазин і товари, що він пропонує. Сайт інтернет-магазину складатиметься з таких функціональних частин: ·Каталог товарів – призначений для процесу впорядкування продукції на сайті інтернет-магазину. Дана функція необхідна для зручного і швидкого пошуку товару. Пошукова система – потрібна для того аби користувач мав змогу швидко знайти потрібну йому інформацію, що особливо важливе у тому випадку, коли є велика кількість розділів, підрозділів і товарів. ·Корзина - призначена для зберігання замовленння користувача. ·Реєстраційна форма - використовується для введення персональних даних користувачів.

·Контактна інформація - є обов'язковою складовою інтернет- магазану, де вказано телефони , e-mail адреси і т.і. Наповнення веб-сайту також включає в себе: пошук потрібної інформації; наповнення інтернет-магазину продукцією; обробку і розміщення фотографій; написання описів до кожного зображення товару; написання описових властивостей до кожного товару. Створенню веб-сайту інтернет-магазину слід приділити увагу його візуальної зручності сприйняття інформації, особливо зручності навігації. Кожна сторінка сайту повинна мати можливість повернення на головну сторінку. Меню повинно бути простим та помітним, щоб користувачу було легко зорієнтуватись на сайті [19].

Контент, є важливим етапом на шляху створення сайту. Правильно підібраний матеріал для інформаційного наповнення гарантує хороший результат.

Уся інформація буде зберигатися в базі даних що створенна за такими принципами:

- гнучкість;
- захищеність;
- адаптивність;
- масштабуємість;
- цілісність.

Інформаційне забезпечення для розроблює мого сайту представлене у вигляді бази даних, що буде містити всю інформацію про товар та його види.

При розробці проекту буде використано наступну інформацію:

- вид продукції;
- одиниці виміру;
- розмір;
- присутність драгоціних камнів;
- тип металу.

Якщо розглядати можливість зростання обсягів даних, то база даних, що використовується, для цієї розробку, з часом буде збільшуватися. . Це

означає, що база даних має розширюватися разом зі збільшенням об’ємів інформації, що вноситься в неї.

Отже, у результаті має бути створена масштабована, надійна та доступна база даних, яка дозволить спростити процеси введення, обробки, аналізу та виведення інформації, що необхідна для проведення щоденної діяльності веб – сторінки.

3.2 Структура та особливості реалізації алгоритмічного забезпечення

Алгоритмічне забезпечення виступає ефективним методом розробки та оптимізації програмного коду у вигляді списку чітко виражених та визначених інструкцій для виконання певної задачі та досягнення поставлених цілей.

Алгоритм – це система правил виконання обчислювального процесу, що обов'язково приводить до розв’язання певного класу задач після скінченного числа операцій. Алгоритмічне забезпечення полегшує процес розробки завдяки структуризації, черговості, поступовому переходу до подальших елементів створення програмного рішення.

При розробці веб – сторінки, що забезпечує взаємодію між користувачем та системаю, було використано мову розмітки HTML та мову опису зовнішнього виду документу CSS.

Під час розробки інтернет – магазину буде використовуватися наступний алгоритм:

1. Складання технічного завдання (ТЗ) .

На першому етапі розробки і створення сайту здійснюється проектування інтерфейсу майбутнього ресурсу і складання технічного завдання. Роботи на цьому етапі багато в чому спираються на ті побажання, які замовник вказав ще при заповненні бріфа, але технічне завдання набагато

детальніше підходить до постановки завдань і детально описує можливості вирішення кожної з них. Не можна недооцінювати всю важливість, яку має ТЗ на розробку сайту для майбутнього ресурсу. Грамотно складене техзавдання дозволяє уникнути непорозуміння між замовником і виконавцем, що в свою чергу економить час, який витрачається на роботу. Тому ТЗ - документ, без якого просто не обійтися. Технічне завдання - це основа порядку в роботі над проектом і головний орієнтир. Тільки з виконання всіх вимог, зазначених у ТЗ, проект може вважатися закінченим.

Свій відбиток у техзавданні знаходять завдання, які ставляться перед ресурсом; докладним чином малюється карта сайту, де вказуються всі розділи та формат інформації, що надається на їхніх сторінках. Велика увага в ТЗ приділяється функціоналу проекту, який описується до дрібниць, а також навігації сайту. Технічне завдання передбачає зазначення вимог до хостингу, верстці, мови програмування і т.п. До всього іншого, ТЗ повинно містити й терміни виконання робіт. Звичайно ж, ця частина технічного завдання прописується, коли описи всіх розділів, функціоналів і вимог сформовані.

2. Розробка дизайну сайту

Як і розробка сайту в цілому, робота над його дизайном також є поетапною. І першим відбувається створення дизайну головної сторінки ресурсу, починається яке з розробки концепції. Вона створюється на основі ТЗ, спроектованого інтерфейсу, заповненого клієнтом брифу і брендбука (при його наявності). За концепцією (або по кожній з них) робляться презентації, метою яких є донести до клієнта причини того чи іншого вибору і рішення, зробленого дизайнерами, пояснити, в чому його вигода для веб-сайту.

Далі відбувається узгодження концепції з замовником та доопрацювання відповідно до зазначених побажаннями клієнта, в ході яких промальовуються способи виділення меню, спливаючі вікна, що випадають списки. Після доопрацювань відбувається затвердження концепції дизайну

головної сторінки, і дизайнери приступають до створення концепцій внутрішніх сторінок, які також проходять етапи узгодження, доопрацювання та затвердження.

Якщо ж компанія-замовник не має логотип - торговий знак, за яким би її ідентифікували, за яким би її дізнавалися, то потрібно розробити його. Природно, що логотип завжди розробляється і узгоджується із замовником впершу чергу, ще до початку відтворення концепції дизайну головної сторінки. Це відбувається тому, що без урахування корпоративної символіки не можна створити по-справжньому оригінальний і цілісний дизайн.

Підсумок етапу «Розробка дизайну сайту» - затверджені макети дизайну всіх сторінок сайту в форматі PSD.

3. Верстка сторінок сайту

Після затвердження макетів дизайну всіх сторінок сайту здійснюється їх верстка. Вебсайт застосовує блочну верстку, так як вона надає великі можливості, ніж таблична, і дозволяє зробити код компактніше, за рахунок чого збільшується швидкість завантаження веб-сторінки. Крім того, блокова верстка дозволяє набагато ефективніше розробляти сайт, який буде коректно відображатися в браузерах.

Верстка здійснюється у відповідності з усіма сучасними вимогами, що пред'являються до неї, проходить перевірку на валідність і сумісність з наступними браузерами:

- Internet Explorer версії 10 та вище;
- Mozila Firefox версії 27.0 та вище;
- Opera версії 9.0 та вище;
- Chrome 37.0 та вище;
- Safari версії 6 та вище.

На етапі верстки здійснюється скриптування елементів дизайну, якщо воно передбачене. Результат, отриманий після закінчення робіт етапу, - зверстані html-шаблони всіх сторінок сайту.

4.Програмування.

На основі ТЗ і верстаючих макетів сторінок сайту здійснюється програмування функціоналу, тобто народжується майже готовий веб-ресурс. Розробка сайту проходить на спеціально відведеному для цих потреб технічному забезпеченні компанії і починається з настроювання середовища розробки (системи управління контентом), де вносяться ті чи інші зміни в залежності від запланованого функціоналу, відбувається програмування необхідних модулів. Результатом етапу є готовий сайт.

Після закінчення робіт з програмування пишеться звіт по роботі з сайтом, в якому клієнту надаються доступи до системи управління сайтом, докладно описується функціонал ресурсу, а також даються рекомендації по роботі з адміністративною частиною сайту.

5.Тестування.

Після того, як роботи зі створення сайту на сервері розробки завершаться, і буде написаний звіт, розпочинається тестування сайту. Для етапу тестування ресурсу передбачена спеціально розроблена методика, за якою і здійснюється перевірка сайту. Перевіряється відповідність сайту описаному в ТЗ функціоналу, коректність відображення верстки у всіх підтримуваних браузерах.

У разі виявлення зауважень, складається перелік доробок, спрямованих на їх усунення. Після цього прийняття проект відправляється на затвердження замовнику. Повторне тестування сайт проходить і після його перенесення на хостинг.

6.Перенесення на хостинг

Заключним етапом розробки сайту є його перенесення на хостинг. Вибор хостингу проходить при участі підприємства – замовника.

3.1 Організаційне забезпечення автоматизованої системи

Для розробки системи було створенно базу даних в котрих містяться 4 таблиці.

	Таблица	Действие	Строки	Тип	Сравнение	Размер	Фрагментировано
☐	category		8	InnoDB	utf8_general_ci	16 КиБ	-
☐	product		56	InnoDB	utf8_general_ci	48 КиБ	-
☐	product_order		2	MyISAM	utf8_general_ci	2.1 КиБ	-
☐	user		2	InnoDB	utf8_general_ci	16 КиБ	-
	4 таблицы	Всего	68	InnoDB	utf8_general_ci	82.1 КиБ	0 Байт

Рисунок 3.1 – Таблиці баз даних

Саме до цих таблиць буде додаватися нові данні, що поступатимуть при створенні нових користувачів, заказів, категорій та додаванні нової продукції.

Детально розлянемо зміст кожної таблиці:

В таблиці представлені «Category» представлені категорії, за допомогою яких відбуватиметься сортування товарів на сайті. В цю таблицю будуть поступати такі данні:

- ID категорії;
- Назва категорії;
- Номер за яким вона буде сортуватися;
- Статус (відображувати на сайті чи ні).

	id	name	sort_order	status
☐	1	Кольца	1	1
☐	2	Серьги	2	1
☐	3	Подвески	3	1
☐	4	Цепочки	4	1
☐	17	Часы	5	1
☐	18	Браслеты	6	1
☐	19	Пирсинг	8	1
☐	20	Уход за украшениями	9	1

Рисунок 3.2 –Данні в таблиці «Category»

В таблиці «Product» будуть зберігатися данні щодо представленої продукції на сайті. Сюди поступатиме такі данні:

- ID продукції;
- Назва товару;
- Ціна;
- Виробник;
- Опис товару;
- Чи новий товар;
- Чи рекомендваний товар;
- Відображається на сайті чи ні;

id	name	category_id	code	price	availability	brand	description	is_new	is_recommended	status
2	Золотое кольцо с фианитом	1	1	1270	1	ООО Закарпатполиметаллы	Для кого: женщине Вид металла: золото Цвет метал...	1	1	1
3	Золотое кольцо с фианитами Swarovski Zirconia	1	0	2324	1	Swarowski	Для кого: женщине Вид металла: золото Цвет...	1	1	1
4	Золотое кольцо «Корона» с фианитами.	1	0	6578	1	Германия		1	1	1
5	Золотое кольцо с алмазной гранью.	1	0	5405	1	Англия	Для кого: женщине Вид металла: золото Цвет...	0	0	1
6	Золотое кольцо с фианитами	1	0	9224	1	Болгария	Стильное эксклюзивное кольцо придется по вкусу ярк...	0	1	1
7	Золотое кольцо с фианитами.	1	0	3732	1	Польша	Кольцо из красного золота 585 пробы с фианитами	0	0	1
8	Золотое кольцо «Корона» с фианитами	1	0	2774	1	Украина	Кольцо из красного золота 585 пробы с фианитами вы...	0	1	1
9	Золотое кольцо с фианитами.	1	0	3254	1	Германия	Золотое кольцо, декорированное россыпью сияющих фи...	0	0	1
10	Золотое кольцо с фианитами.	1	0	4315	1	Чехия	Кольцо из красного золота 585 пробы с фианитами	0	0	1
11	Золотое кольцо с алмазной гранью.	1	0	2540	1	Греция	Кольцо из красного золота 585 пробы с алмазной гра...	0	0	1
12	Maurice Lacroix	17	0	7560	1	Швейцария	ретроградный указатель дня недели, 22 камня в меха...	0	0	1
13	Longines	17	0	10499	1	Швейцария	Серия: Presence Водостойкость: 30 Ремешо...	1	0	1
14	Maurice Lacroix	17	0	7800	1	Швейцария	Серия: Aurea Страна производитель: Швейцария<...	0	1	1
15	Louis Erard	17	0	5400	1	Швейцария	Водостойкость: 50 Календарь: дата Размер...	1	0	1
16	Omega	17	0	2500	1	Швейцария	Серия: Constellation Страна производитель: Шв...	0	1	1
17	Omega	17	0	4500	1	Швейцария	Водостойкость: 100 Календарь: дата Разме...	1	0	1

Рисунок 3.3 – Данні в таблиці «Product»

В таблиці «Product_order» зберігатимуться данні про створені закази на сайті, тому сюди поступатимуть дані о ID заказу, ім’я користувача, телефон користувача, дату заказу, продукцію що заказали, статус заказу, коментарій заказчика та ID юзера.

id	user_name	user_phone	user_comment	user_id	date	products	status
48	Admin	56456456443334		1	2017-12-21 11:51:05	{"43":1}	1

Рисунок 3.4 – Данні в таблиці «Product_order»

В таблиці «User» зберігаються інформацію про створені аккаунти та їх інформцію, а саме:

- ID аккаунта;
- Назва аккаунту;
- email;
- Пароль;
- та роль в системі.

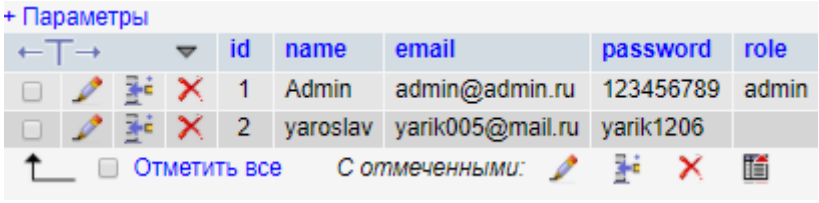


Рисунок 3.5 –Данні в таблиці «User»

Пілся створення таблиць баз данних, використовуючи язик PHP, HTML, CSS потрібно створити сайт, котрий буде взаємодіяти з базою данних.

3.1.1 Інструкція для працівника

Так як нова конфігурація буде в новинку користувачам заводу, знадобиться багато часу та додаткові фінансові вкладення в адаптацію продукту серед співробітників. Для вирішення такого роду завдання, допоможе розробка інструкції користувача, куди буде розміщено детальний опис кожного пункту та візуальне представлення автоматизованої системи.

Щоб уникнути небажаних змін данних потрібен входу в систему з акантом котрому надаються особливі права. Логін має формау admin@admin.ru, пароль має мінімум 8 символів.

Для початку потрібно натиснути кнопку «Вход» рис. 3.1.

Рисунок 3.6 – Кнопка «Вход»

Після чого попадаємо на сторінку де потрібно ввести логін та пароль рис. 3.2.

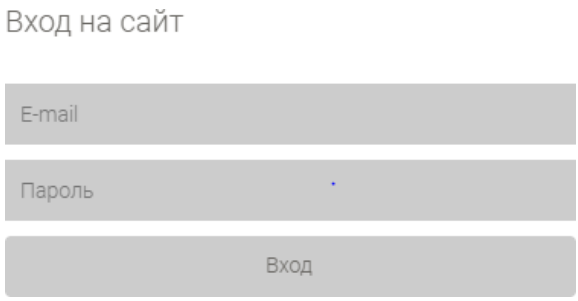


Рисунок 3. 7 – Форма логіну та пароля

Після цього в адресну строку потрібно ввести данні панелі адміністратора рис. 3.3.

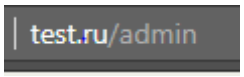


Рисунок 3.8 – Вхід до панелі адміністратора

Після введення логіну та паролю працівник потрапляє в панель адміністрування, де можна спостерігати три функції адміністрування:

- управление товарами;
- управление категориями;
- управление заказами.

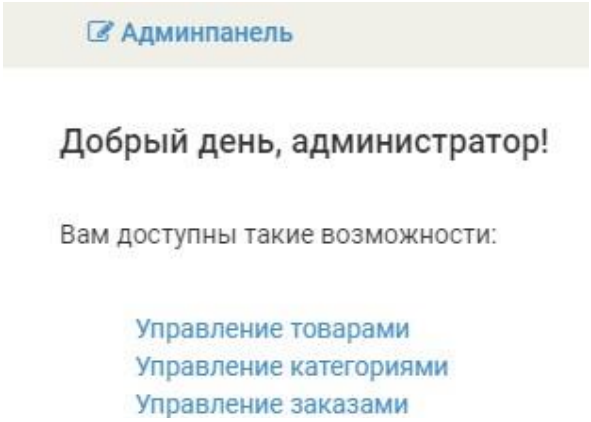


Рисунок 3.9 – Панель адміністрування

По-перше розпочнемо з першої вкладки, яка з’являється відразу після авторизації співробітника – «Управление товарами».

Тут відкриваються нашому зору короткі дані про наявність товару, його назву, ID товару, артикул, ціну.

За допомогою цієї вкладки користувач може переглянути інформацію про товар та змінити її, натиснувши кнопку «Редактировать», або зовсім видалити її, натиснувши кнопку «Удалить».

В цьому вікні також можливо додавати новий товар на продаж, натиснувши кнопку «Добавить товар», рис. 3.5.

+ Добавить товар

Список товаров

ID товара	Артикул	Название товара	Цена		
2	1	Золотое кольцо с фианитом	1270		
3	0	Золотое кольцо с фианитами Swarovski Zirconia	2324		
4	0	Золотое кольцо «Корона» с фианитами.	6578		
5	0	Золотое кольцо с алмазной гранью.	5405		
6	0	Золотое кольцо с фианитами	9224		
7	0	Золотое кольцо с фианитами.	3732		
8	0	Золотое кольцо «Корона» с фианитами	2774		
9	0	Золотое кольцо с фианитами.	3254		
10	0	Золотое кольцо с фианитами.	4315		
11	0	Золотое кольцо с алмазной гранью.	2540		
12	0	Maurice Lacroix	7560		
13	0	Longines	10499		
14	0	Maurice Lacroix	7800		
15	0	Louis Erard	5400		
16	0	Omega	2500		
17	0	Omega	4500		
18	0	Conteri	7500		
19	0	Zerconi	5400		

Рисунок 3.10 – «Управление товарами»

Натиснувши кнопку «Добавить товар» ми опинимося в підменю, де маємо можливість добавити новий товар на продаж.

Перш ніж добавити товар на продаж, користувач повинен ввести основні характеристика товару, такі як:

- «Название товара»;
- «Артикул»;
- «Стоимость»;
- «Категория»;
- «Производитель»;

- «Изображение товара»;
- «Детальное описание»;
- «Наличие на складе»;
- «Новинки»;
- «Рекомендуемые»;
- «Статус».

Меню додавання новго товару наведено на риснку 3.6.

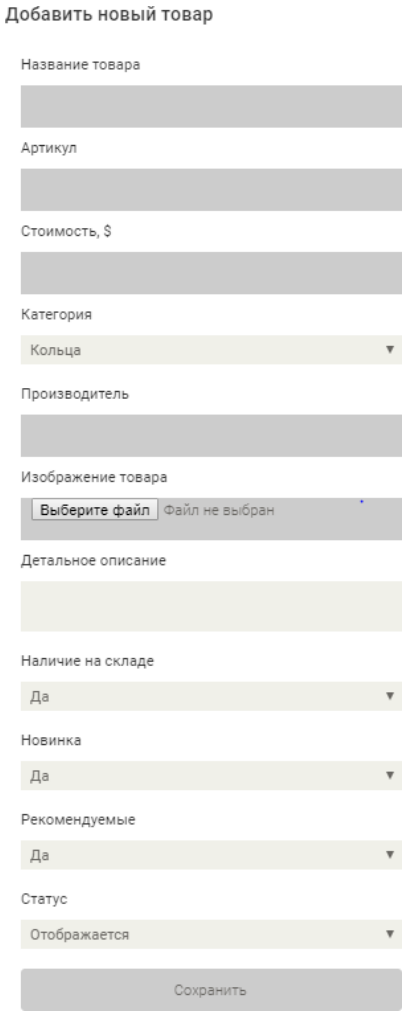


Рисунок 3.11 – Меню додавання новго товару

Деякі поля мають обмеження на внесення даних. В полі артикул потрібно вводити дані, за котрими можна знайти саме цей товар посеред інших. Ці данні повинні бути унікальні для кожного із товарів.

Строка «Стоимость» вводиться тільки у числовому виглід, без припису грошового еквіваленту. В стандарті використовується гривня, як грошовий еквівалент.

При виборі варіанту «Да» у пунктах «Новинка» та «Рекомендуемые» товар відображатиметься у каталозі з спеціальними лейблами , рисунок 3.7.

Рисунок 3.12 – Лейбли товарів

Характеристика «Статус» визначає відображувати товар у каталозі чи ні.

«Изображение товара» - потрібно додати файл, за допомогою кнопки «Выберите файл». Файл повинен бути не більше 300x300px, та мати розширення jpg або ж png.

Після того, як користувач введе всі характеристики товару потрібно натиснути кнопку «Сохранить» і данний товар одразу після натискання опинеться на сторінці Інтернет – магазину для продажу.

Переходимо до наступної вкладки головного меню «Управление категориями», де розташовані категорії товарів для продажу. (рис. 3.8).

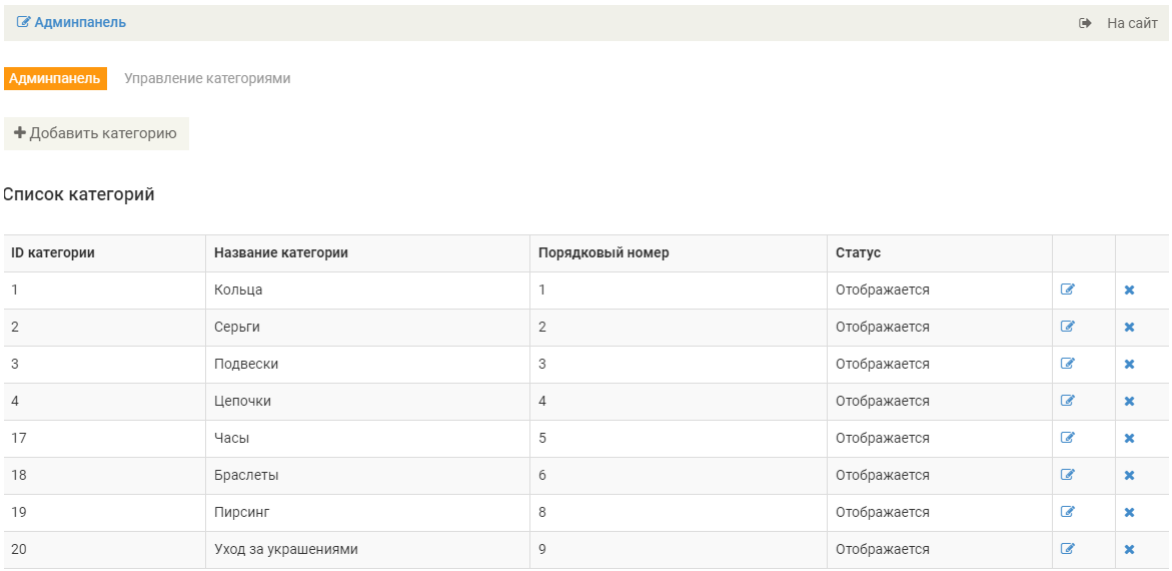


Рисунок 3.13 – Управление категориями

У цьому розділі ми можемо бачити данні про категорії товарів на сайті. А саме, «ID категории», «Название категории», «Порядковый номер», «Статус».

«ID категории» має унікальне числове значення для кожної категорії, що дає змогу визначити окрему категорію серед інших. «Название категории» визначає назву окремої категорії. «Порядковый номер» - це числове значення від 1 до 100, саме ці значення дають змогу відображати одну категорію вище іншох в каталозі. Характеристика «Статус» показує значення від якого залежить відображається чи ні категорія на сайті.

Саме у цьому меню ми маємо можливість змінювати, створювати та видаляти категорії товарів, за якими буде відсортовано товари в каталозі. Щоб видалити категорію товару потрібно натиснути кнопку «Удалить», а для того щоб внести зміни до категорії потрібно натиснути «Редактировать».

Щоб додати нову категрію до каталоги, по-перше потрібно натиснути кнопку «Добавить категорію» (рис.3.9) . Після чого користувач потрапить до іншої сторінки, де матиме змогу стоворити категорію.

Для цього йому потрібно ввести характеристика категорії що він хоче додати, а саме:

- «Название»;

- «Порядковый номер»;
- «Статус».

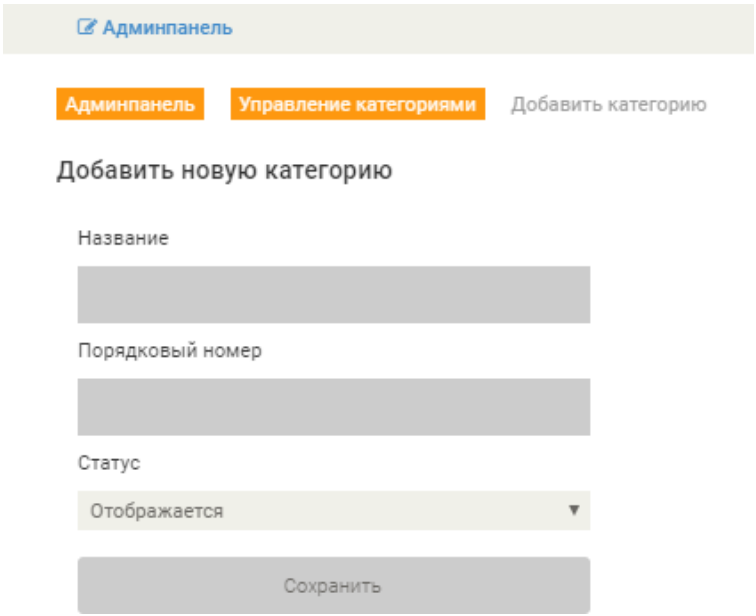


Рисунок 3.14 – Додавання нової категорії

В полях потрібно ввести назву категорії, її порядковий номер від 1-100, статус, після чого натиснути кнопку «Сохранить». Після цього для категорії автоматично згенерується «ID номер» і вона зьявиться в каталозі сайту.

Наступним розгялнмо інший пунк миеню, а саме: Управление заказами (рис.3.10).

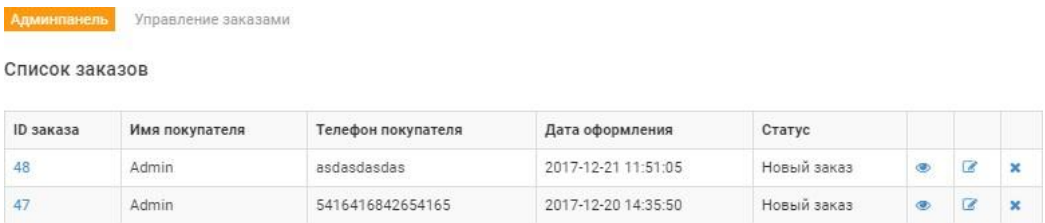


Рисунок 3.15 – «Управление заказами»

В цьому меню здійсньється обробка заказів клієнтів. Саме тут можно побачити «ID заказу», «Имя покупателя», «Телефон покупателя», «Дата

формления», «Статус». Ця сторінка є основною для взаємодії з заказами та інформацією клієнтів.

«ID заказу» відображає порядковий номер заказу, «Имя покупателя» - зазначене при заказі Ім’я, на котре буде зарезервований товар до зв’язку з опереатором та відвантажений товар. «Телефон покупателя» зазначається у цифровому вигляді для зв’язка оператора та покупця. «Телефон покупателя» показує час підтвердження заказу. «Статус» визначає вид заказу, він може містити такий статус:

- «Новий заказ»;
- «В оброботке»;
- «Доставляеться»;
- «Закрыт».

Також тут можливо видалити заказ натиснувши «Удалить»,після ого потрібно підтвердити видалення заказу (рис.3.11).

Рисунок 3.16 – Видалення заказу

Редагування заказу можливо натиснувши «Редактировать», після чого користувач опиняється іншому меню де він може редагувати заказ (рис. 3.12).

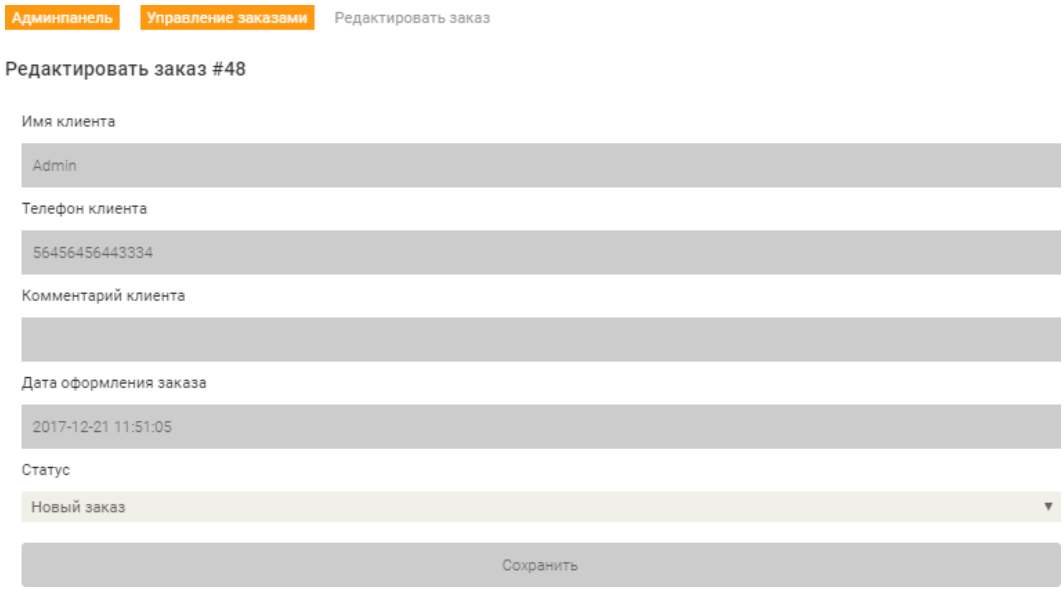


Рисунок 3.17 – Редагування заказу

Щоб підтвердити зміни у заказі потрібно натиснути «Сохранить».

В «Управление заказами» також можливо подивитися всю інформацію о зказі натиснувши кнопку «Смотреть» (рис. 3.13). В цьому меню відображається інформація про заказ, клієнта, статус заказа та товар і кількість його у заказі.

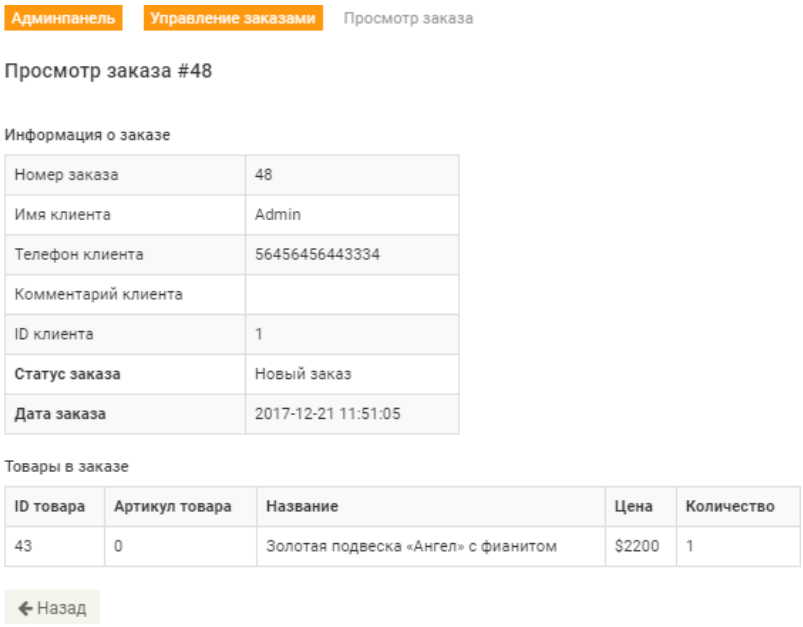


Рисунок 3.18 – Просмотр інформації про заказ

Після завершення роботи в системі потрібно натиснути кнопку «Выход» для запобіганню несанкціановоного доступу до системи.

3.1.2 Інструкція для покупця

Проект автоматизації підприємства розрахован на використання не тільки працівниками а й клієнтами, тому розроблений проект дозволяю клієнтам здійснювати покупки не виходячи з дому у інтернет – магазині.

За для зручного користуванням клієнтом, було розроблено просту у використанні систему. Для того щоб почати роботу, клієнту потрібно зайти на сайт інтернет – магазину, що має вигляд який наведено на рисунку 3.1.:

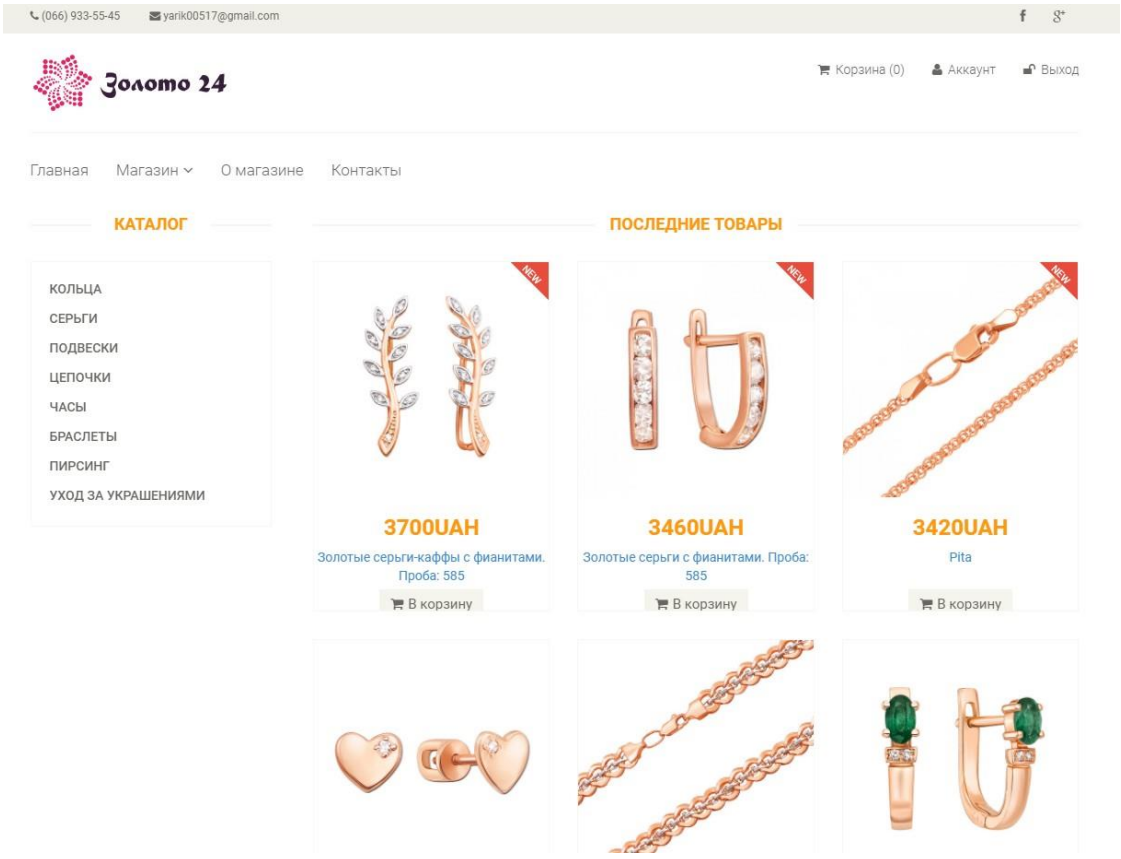


Рисунок 3.19 – Головна сторінка

В самому початку сторінки є контактна інформація магазину, де можна отримати відповіді на запитання щодо товару та обслуговування.

На головній сторінці можна побачити останні товари, що були добавленні до катологу, також є можливість добавити їх одразу до корзини.

Головна сторінка має багато меню та фільтрів для зручного використання. Клієнт може відфільтрувати товар за наступними видами:

- «Кольца»;
- «Серьги»;
- «Подвески»;
- «Цепочки»;
- «Часы»;
- «Браслеты»;
- «Пирсинг»;
- «Уход за украшениями».

Використовуючи ці фільтри клієнт має можливість швидко знайти каталог товарів за видом, що він шукає.

В самому низу сторінці можно побачити товари що підпадають під акційні пропозиції.

Також на головній сторінці є декілька ссилок на інші сторінки, в котрих можно більше дізнатися про магазин та каталог товарів, а саме:

- «О Магазине»
- «Контакты»

В вкладці «О Магазине» можливо дізнатися основну інформацію про магазин, його політику та походження товару.

Дані наведено на рисунку 3.15.

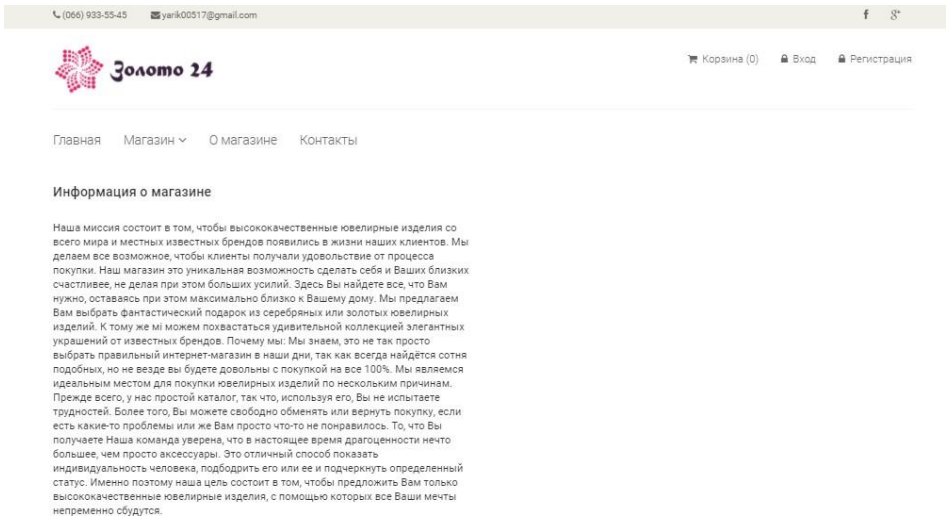


Рисунок 3.20 – Основна інформація про магазин

А вже в меню «Контакти» (рис.3.16) клієнт може дізнатися часи роботи магазину, місце знаходження його представництва, та інші методи зв’язку з магазином.

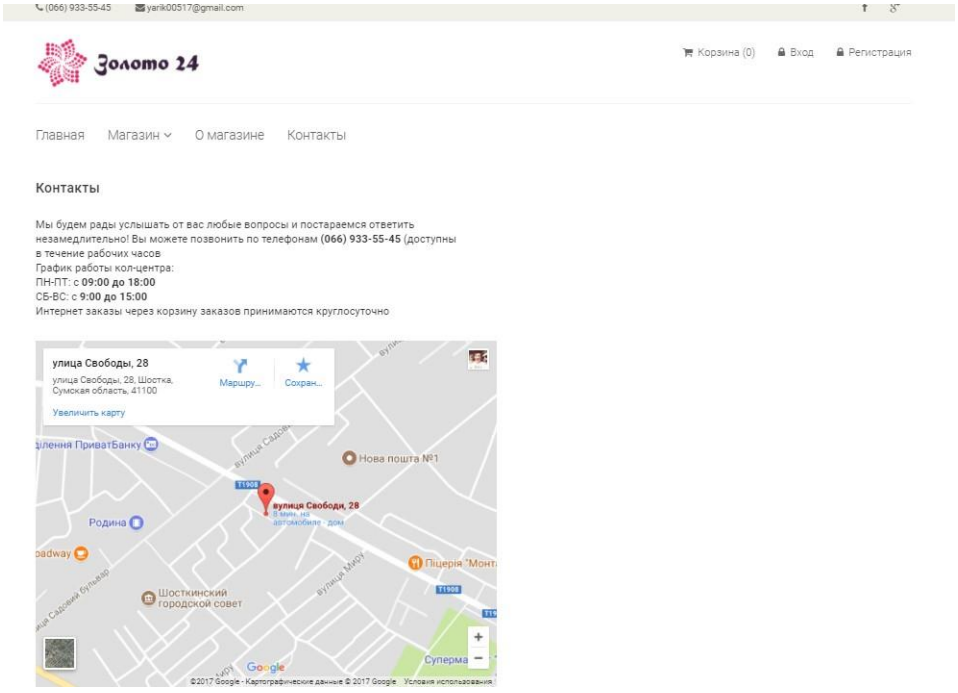


Рисунок 3.21 – Контактна інформація

Для повернення на головну сторінку магазину було розробленно декілька методів. Натиснувши ссилку «Главная» або на емблему магазину клієнт потрапить на головну сторінку, де він може продовжити вибор товарів за його розсудом.

Після того, як клієнт знайте вподобаний йому товар він повинен натиснути кнопку «В корзину», що добавить вибраний їм товари до віртуальної корзини та дозволить зробити на нього замовлення. Якщо, людина бажає зробити не одне замовлення він може повторно використовувати «В корзину» на інших товарах (рис. 3.17).

Вибравши товар, для замовлення клієнт повинен перейти до віртуальної корзини.

Рисунок 3.22 – Кнопка переходу до корзини

В цьому меню буде відображатися товар, який людина добавила до замовлення власноруч (рис.3.18).

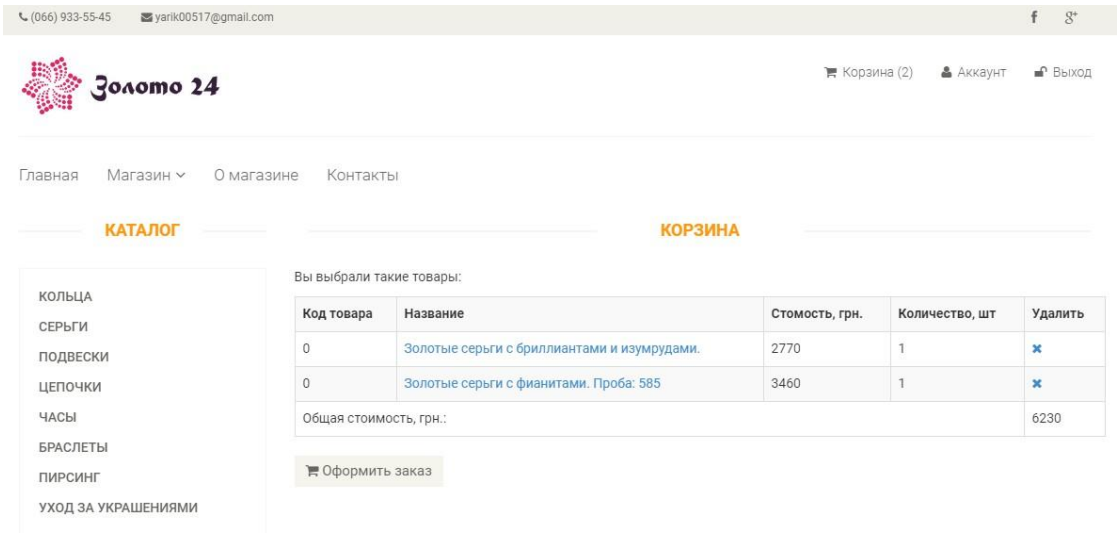


Рисунок 3.23 – Віртуальна корзина

На цій сторінці людина може подивитися підсумкове замолення, назву товару, ціну товарів, як окрему так і загальну. В віртуальній корзині можливо видалити раніше обраний товар, для цього потрібно натиснути кнопку «Удалить» напроти зайвого товару.

Для подальшого оформлення товару потрібно натиснути кнопку «Оформить заказ», що презведе до переходу на іншу сторінку (рис.3.19).

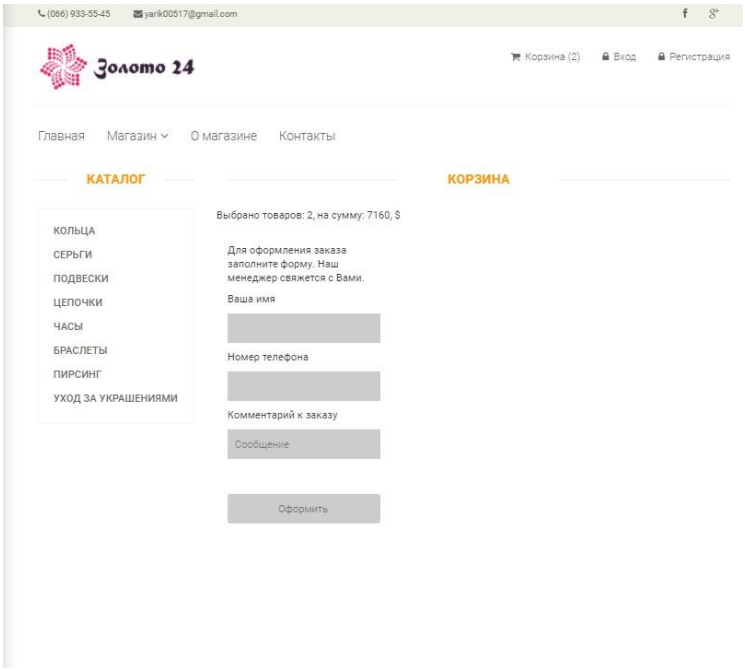


Рисунок 3.24 – Сторінка підтвердження заказу

На цій сторінці клієнт повинен заповнити деякі поля, а саме:

- «Ваше имя»
- «Номер телефона»
- «Коментарий к заказу»

Указавши ім’я клієнту, номер телефону та коментарі до заказу потрібно натиснути кнопку «Оформить», після чого заказ перейде до обробки менеджерами. Через деякий час менеджер зв’яжиться з клієнтом для обговорення остаточної інформації.

Для зручності клієнтів було також розроблено підсистему регестрації клієнтів. Ця система дозволяє постійним клієнтам не витрачати час для ведення інформації про нього.

Для здійснення регестрації, на головній сторінці потрібно натиснути кнопку «Регестрация». Це презведе для переходу клієнта на іншу сторінку. Форму регестрації наведено на рисунку 3.20.

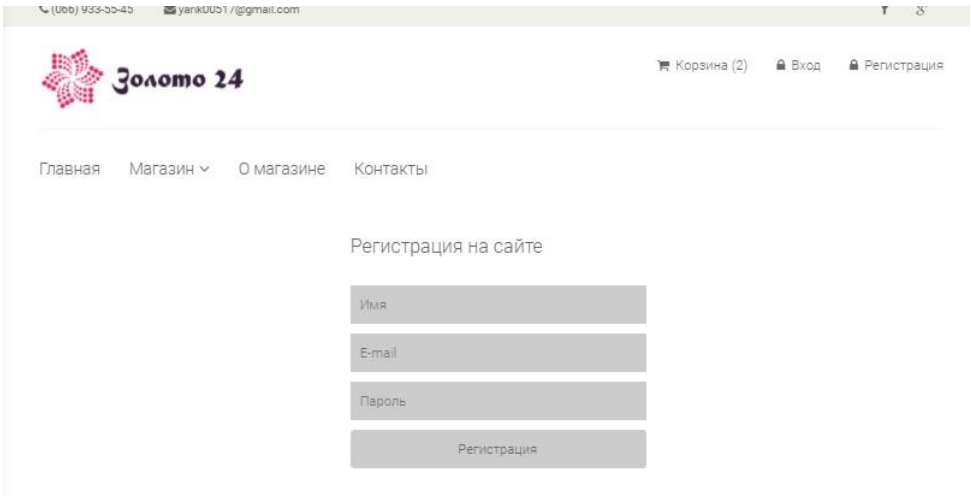


Рисунок 3.25 – Форма регестрації на сайті

На цій сторінці клієнт повинен вказати своє ім’я, електрону адресу та пароль, після чого натиснути кнопку регестрації. Зареєструвавшись клієнт зможе війти в систему для більш швидких покупок у майбутньому.

3.2 Оцінка економічного ефекту від впровадження автоматизованої системи

Ефективність — це економічна категорія, що відображає співвідношення між одержаними результатами і витраченими на їх досягнення ресурсами, причому при вимірюванні ефективності ресурси можуть бути представлені або в певному обсязі за їх первісною вартістю, або частиною їх вартості у формі виробничих витрат.

Економічний ефект від впровадження інформаційних систем поділяють на прямий і непрямий.

Автоматизація продажі та обліку товарів здатне змінити час на обслуговування клієнтів, зменшити затрати на заробітню плату, збільшити кількість продажів.

Пряма економічна ефективність - економія матеріально-трудових ресурсів та грошових засобів, отримана в результаті скорочення чисельності управлінського персоналу, фонду заробітної плати, витрат основних і допоміжних матеріалів внаслідок автоматизації конкретних видів плановообліkових і аналітичних.

Для впровадження програмного продукту потрібно понести певну суму затрат. За допомогою економічних показників було розраховано пряму ефективність та термін окупності.До впровадження автоматизації усі операції по веденю звітності скалду велися вручну, тому за допомогою впровадження було зменшено кількість часу, що відводилося для ведення звітності.

Непрямий ефективність показала переваги котрі надала автоматизація підприємству:

- збільшення клієнтів;
- збільшення продуктивність співробітників;
- скорочення часу на ведення звітності складу;
- підвищення швидкості пошуку інформації.

Вирахувавши ефективність підприємство вмає можливість вирішити подальшу доцільність додавання нових розробок і функцій у автоматизовану систему.

Розрахуємо економічний ефект від впровадження розробленої системи на об’єкті автоматизації табл.3.1 Перш за все необхідно зазначити, що впроваджуватиметься лише програмне забезпечення, оскільки наявне на об’єкті апаратне забезпечення дає можливість ефективного функціонування розробленої системи. Тому заміна апаратного забезпечення не потрібна, що значно зменшує витрати на впровадження системи.

Для розрахунку річного ефекту та терміну окупності системи використаємо умовні позначення, наведені у таблиці.

Таблиця 3.1 – Основні позначення розрахунку ефективності системи

№	Позначення	Пояснення
1	Впроект	Витрати на проектування комплексу задач
2	Впрогр	Витрати на програмування комплексу задач
3	Вн	Витрати на програмування в період налагодження ПЗ комплексу задач
4	Ввпров	Витрати на впровадження
5	Вб	Приведені до одного року витрати на обробку інформації при базовому варіанті організації обробки
6	Вп	Приведені до одного року витрати на обробку інформації при впроваджуваному (пропонованому) варіанті організації системи обробки
7	E_y	Річний економічний ефект
8	S	Річна економія, яку отримає підприємство в результаті впровадження АСУ
9	C	Капітальні вкладення, які було витрачено в результаті впровадження АСУ
10	r_n	Нормативний коефіцієнт окупності капітальних вкладень, узятий для конкретної галузі

Формули для розрахунку та результати наведені у таблиці 3.2

Таблиця 3.2 – Розрахунок ефективності системи

№	Формула	Опис формули	Розрахунок
1	С = Впроект + Впрограм + Вн + Ввпров	Сума капітальних витрат	5840
2	S=Вб–Вп	Сума річної економії	9450 – 3780 = 5760
3	E =S–C×r	Річний ефект від впровадження	5760 – 1752 = 4008
4	P = C/S = 1/Rce	Термін окупності	5840 / 5760=1,01 р.

Розрахувавши ефективність створеної системи, можна стверджувати, що вона позитивно впливатиме на витрати підприємства і буде оптимізувати його діяльність.

ВИСНОВКИ

Під час виконання випускної роботи було розглянуто та проаналізовано діяльність підприємства. Було виконано постонову задачі, основних вимог до програмного продукту з метою розробки автоматизованої системи для цього підприємства.

До автоматизації процес організації продажу та обліку товарів підприємство використовує переважно засоби табличних редакторів, а у багатьох випадках проведення обіку наявності товарів проводиться вручну.

Успішний розвиток будь-якого підприємства неможливо без автоматизації, яка грунтується на використанні передових інформаційних технологій. Тому під час роботи було розглянуто багато варіатів автоматизації на даному підприємстві, виділивши плюси та мінуси всіляких методів автоматизації було досягнуто рішення розробити веб сторінку з можливістю реалізовувати товар в мережі інтернет, а також можливістю обліку наявності товарів на данному підприємстві.

Було проаналізовану технічні та інформаційні потоки підприємства, також був розроблений алгоритм за яким буде розроблятися автоматизація саме для цього підприємства.

Було описано апаратні рішення, які мінімально необхідні задля правильного та довгово використання системи.

В ході дослідження були виконані всі поставлені задачі та проаналізовано напрями даної системи. Виконано постановку задачі та визначено основні вимоги до проектування програмного продукту з метою подальшої розробки системи. Наступним етапом стало обрання технології, яка успішно вирішить дану задачу. Після порівняння та аналізу існуючих технологій було обрано сучасні методи рішення поставлених задач. Для створення автоматизованої системи були використані :

– HTML5

- CSS3
- SQL
- Bootstrap
- PHP
- JavaScript

Також будуть застосовуватися програмні засоби для рішення поставлених проблем:

- Sublime Text
- WebShtorm
- OpenServer

Детально було описано рекомендоване для підприємства апаратне та програмне забезпечення, що забезпечить успішне функціонування автоматизованої системи протягом довгого часу. В деталях описано процес конфігурування автоматизованої системи, візуальне представлення та розроблено інструкцію по використанню розробленого прототипу автоматизованої системи. За допомогою впровадженого програмного продукту всі операції будуть реалізовані набагато швидше ніж раніше. В результаті підприємство отримає економію за за рахунок скорочення кількості операцій, та можливості продажу товарів цілодобово.

Структура бази даних складається з 4 таблиць, які пов’язані між собою зв’язком типу один до багатьох, що забезпечує збереження цілісностіданих. Структура Web-сайту базується на адміністративній та клієнтській частині.

Якщо розглядати ефективність даного рішення автоматизації, то варто зазначити, що його використання надасть змогу скоротити час на обслуговування клієнтів та проводити менше ручної роботи.

Даний прототип програмного продукту автоматизації механізму клієнтського обслуговування та обліку товарів на підприємстві в подальшому має всі перспективи для розвитку. В майбутньому можна додати функціонал для відстеження статистики покупок та впровадження системи бонусів або

скидок, а також розробка системи оповіщення постійних клієнтів за допомогою електроної пошти і розробки додатку для мобільних платформ.

Сильними сторонами розробленого прототипу веб –сайту є:

1. Зручний інтерфейс користувача;

2. Сортування товарів по групах, що пришвидшує пошук потрібного товару;

3. Автоматична зміна даних в базі даних магазину при здійсненні замовлення обраного товару.

Негативними якостями Інтернет-магазину в цьому стані є :

1. Відсутність можливості оплати новітніми засобами;

2. Ціни товарів не представленні в іноземній валюті.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Автоматизация торговли. Основные правила и аспекты. Автоматизация учёта на предприятии. [Електронний ресурс] – Режим доступу до ресурсу: https://expresservice.com.ua/news/avtomatizacija_torgovli_magazina.

2. Автоматизация учета в розничной торговле. Оборудование и программное обеспечение [Електронний ресурс] – Режим доступу до ресурсу: https://www.business.ru/article/525-avtomatizatsiya-uchta-v-roznichnoy-torgovle-oborudovanie-i-programmnoe-obespechenie.

3. Автоматизація [Електронний ресурс] – Режим доступу до ресурсу:https://uk.wikipedia.org/wiki/%D0%90%D0%B2%D1%82%D0%BE%D0%BC%D0%B0%D1%82%D0%B8%D0%B7%D0%B0%D1%86%D1%96%D1%8F.

4. Адаптивный, отзывчивый, резиновый дизайн - что это такое? [Електронний ресурс] – Режим доступу до ресурсу: http://wseweb.ru/diz/adaptivny-dizain.htm.

5. Астахова, И.Ф. SQL в примерах и задачах [Текст] : учебн. пособие И. Ф. Астахова, А. П. Толстобородов, В. М. Мельников – Минск : Новое знание, 2008. – 176 с. – ISBN 985-475-004-3.

6. Бойцы невидимого фронта [Електронний ресурс] – Режим доступу до ресурсу: http://citforum.ru/internet/webd/article_19.shtml.

7. Веб-программирование (обзорная статья) [Електронний ресурс] – Режим доступу до ресурсу: http://wseweb.ru/diz/obzor3.htm.

8. Гагарина, Л. Г. Разработка и эксплуатация автоматизированных информационных систем [Текст]: учеб. пособие / Л. Г. Гагарина, Д. В. Киселев, Е. Л. Федотова ; [под ред. проф. Л. Г. Гагариной]. – Москва : ИД «Форум» : ИНФРА–М, 2009. − 384 с.

9. Глазок, О.М. Застосування інформаційних технологій в

оптимізації роботи бізне-структур: [Електронний ресурс]. − Режим доступа: http://avia.nau.edu.ua/doc/2011/5/avia2011_5_10.pdf. – 10.06.2015. – Заголовок с экрана.

10. ГОСТ 12.1.030 - 81*. ССБТ. Электробезопасность. Защитное заземление. Зануление. - Введ. 01.01.82/

11. Как создать привлекательный web-сайт? [Електронний ресурс] – Режим доступу до ресурсу: http://citforum.ru/internet/webd/article_1.shtml.

12. Кит Томсон. Автоматизация продаж. Умный подход / Т. Томсон Кит., 2008. – (Вершина).

13. Обзор возможностей интегрированной среды разработки Visual Studio [Електронний ресурс] – Режим доступу до ресурсу: https://docs.microsoft.com/ru-ru/visualstudio/ide/visual-studio-ide.

14. Основные ошибки при открытии небольшого Интернет-магазина [Електронний ресурс] – Режим доступу до ресурсу: https://habrahabr.ru/company/Centrobit/blog/146011/.

15. Особливості автоматизації обліку товарообігу в сучасних умовах [Електронний ресурс] – Режим доступу до ресурсу: https://www.google.com.ua/url?sa=t&rct=j&q=&esrc=s&source=web&cd=2&ved=0ahUKEwiT1LbTvK_YAhXK16QKHRSZAdIQFgg7MAE&url=http%3A%2F%2Fwww.irbis-nbuv.gov.ua%2Fcgi-bin%2Firbis_nbuv%2Fcgiirbis_64.exe%3FC21COM%3D2%26I21DBN%3DUJRN%26P21DBN%3DUJRN%26IMAGE_FILE_DOWNLOAD%3D1%26Image_file_name%3DPDF%2FZnpdetut_eiu_2013_25_42.pdf&usg=AOvVaw3ixyiGvS4jTew-zH7UK3YO.

16. Правила охорони праці під час експлуатації електронно - обчислювальних машин . №382/3675

17. Практическое руководство. Перемещение по интегрированной среде разработки Visual Studio [Електронний ресурс] – Режим доступу до ресурсу: https://docs.microsoft.com/ru-ru/visualstudio/ide/how-to-move-around-in-the-visual-studio-ide.

18. Программы, используемые в web-мастеринге [Електронний ресурс] – Режим доступу до ресурсу: http://citforum.ru/internet/webd/article_3.shtml.

19. С чего начинается сайт? [Електронний ресурс] – Режим доступу до ресурсу: http://citforum.ru/internet/webd/article_28.shtml.

20. Учебник HTML [Електронний ресурс] – Режим доступу до ресурсу: http://ru.html.net/tutorials/html/.

21. Что такое CSS? [Електронний ресурс] – Режим доступу до ресурсу: http://ru.html.net/tutorials/css/lesson1.php.

22. Что такое PHP? [Електронний ресурс] – Режим доступу до ресурсу: http://php.net/manual/ru/intro-whatis.php.

23. Языки програмирования Регістри відомостей [Електронний ресурс]: − Режим доступу : http://life-prog.ru/ukr/1_2187_registrividomostey.html/ – 10.06.2015 – Назва з екрану.

24. Bootstrap - About [Електронний ресурс] – Режим доступу до ресурсу: https://getbootstrap.com/docs/4.0/about/overview/.

25. Faster Rails Development with Zeus [Електронний ресурс] – Режим доступу до ресурсу: https://confluence.jetbrains.com/display/RUBYDEV/Faster+Rails+Development+with+Zeus.

26. Guided Tour Around WebStorm User Interface [Електронний ресурс] – Режим доступу до ресурсу: https://www.jetbrains.com/help/webstorm/guided-tour-around-webstorm-user-interface.html.

27. How to start Ruby on Rails development in RubyMine on Windows [Електронний ресурс] – Режим доступу до ресурсу: https://confluence.jetbrains.com/display/RUBYDEV/How+to+start+Ruby+on+Rails+development+in+RubyMine+on+Windows.

28. HTML [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/HTML.

29. HTML [Електронний ресурс] – Режим доступу до ресурсу: https://www.jetbrains.com/help/webstorm/html.html.

30. Installing and Upgrading MySQL [Електронний ресурс] – Режим доступу до ресурсу: https://dev.mysql.com/doc/refman/5.7/en/installing.html.

31. JavaScript и объектная модель [Електронний ресурс] – Режим доступу до ресурсу: http://citforum.ru/internet/webd/article_22.shtml.

32. JavaScript: полезные функции [Електронний ресурс] – Режим доступу до ресурсу: http://citforum.ru/internet/webd/article_29.shtml.

33. MySQL 5.7 Reference Manual [Електронний ресурс] – Режим доступу до ресурсу: https://dev.mysql.com/doc/refman/5.7/en/.

34. Sublime Text Documentation [Електронний ресурс] – Режим доступу до ресурсу: https://www.sublimetext.com/docs/3/.

35. Visual Studio Samples [Електронний ресурс] – Режим доступу до ресурсу: https://docs.microsoft.com/ru-ru/visualstudio/ide/visual-studio-samples.

ДОДАТКИ

Додаток А

Таблиці баз данних

	Таблица	Действие	Строки	Тип	Сравнение	Размер	Фрагментировано
☐	category		8	InnoDB	utf8_general_ci	16 КиБ	-
☐	product		56	InnoDB	utf8_general_ci	48 КиБ	-
☐	product_order		2	MyISAM	utf8_general_ci	2.1 КиБ	-
☐	user		2	InnoDB	utf8_general_ci	16 КиБ	-
	4 таблицы	Всего	68	InnoDB	utf8_general_ci	82.1 КиБ	0 Байт

Рисунок А.1 – Основні таблиці бази даних

	id	name	sort_order	status
☐	1	Кольца	1	1
☐	2	Серьги	2	1
☐	3	Подвески	3	1
☐	4	Цепочки	4	1
☐	17	Часы	5	1
☐	18	Браслеты	6	1
☐	19	Пирсинг	8	1
☐	20	Уход за украшениями	9	1

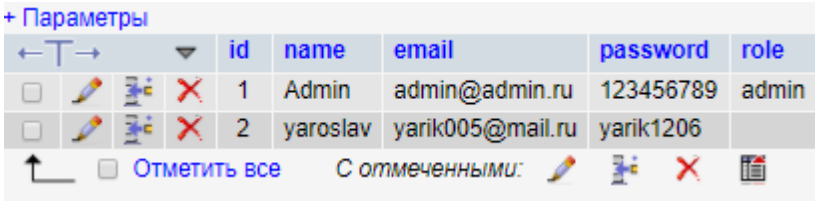
Рисунок А.2 – Данні в таблиці «Category»

id	name	category_id	code	price	availability	brand	description	is_new	is_recommended	status
2	Золотое кольцо с фианитом	1	1	1270	1	ООО Закарпатполиметаллы	Для кого: женщине Вид металла: золото Цвет метал...	1	1	1
3	Золотое кольцо с фианитами Swarovski Zirconia	1	0	2324	1	Swarowski	Для кого: женщине Вид металла: золото Цвет...	1	1	1
4	Золотое кольцо «Корона» с фианитами.	1	0	6578	1	Германия		1	1	1
5	Золотое кольцо с алмазной гранью.	1	0	5405	1	Англия	Для кого: женщине Вид металла: золото Цвет...	0	0	1
6	Золотое кольцо с фианитами	1	0	9224	1	Болгария	Стильное эксклюзивное кольцо придется по вкусу ярк...	0	1	1
7	Золотое кольцо с фианитами.	1	0	3732	1	Польша	Кольцо из красного золота 585 пробы с фианитами	0	0	1
8	Золотое кольцо «Корона» с фианитами	1	0	2774	1	Украина	Кольцо из красного золота 585 пробы с фианитами вы...	0	1	1
9	Золотое кольцо с фианитами.	1	0	3254	1	Германия	Золотое кольцо, декорированное россыпью сияющих фи...	0	0	1
10	Золотое кольцо с фианитами.	1	0	4315	1	Чехия	Кольцо из красного золота 585 пробы с фианитами	0	0	1
11	Золотое кольцо с алмазной гранью.	1	0	2540	1	Греция	Кольцо из красного золота 585 пробы с алмазной гра...	0	0	1
12	Maurice Lacroix	17	0	7560	1	Швейцария	ретроградный указатель дня недели, 22 камня в меха...	0	0	1
13	Longines	17	0	10499	1	Швейцария	Серия: Presence Водостойкость: 30 Ремешо...	1	0	1
14	Maurice Lacroix	17	0	7800	1	Швейцария	Серия: Aurea Страна производитель: Швейцария<...	0	1	1
15	Louis Erard	17	0	5400	1	Швейцария	Водостойкость: 50 Календарь: дата Размер...	1	0	1
16	Omega	17	0	2500	1	Швейцария	Серия: Constellation Страна производитель: Шв...	0	1	1
17	Omega	17	0	4500	1	Швейцария	Водостойкость: 100 Календарь: дата Разме...	1	0	1

Рисунок А.3 – Данні в таблиці «Product»

id	user_name	user_phone	user_comment	user_id	date	products	status
48	Admin	56456456443334		1	2017-12-21 11:51:05	{"43":1}	1

Рисунок А.4 – Данні в таблиці «Product_oreder»



+ Параметры

id	name	email	password	role
1	Admin	admin@admin.ru	123456789	admin
2	yaroslav	yarik005@mail.ru	yarik1206	

Отметить все С отмеченными:

Рисунок А.5 – Данні в таблиці «User»

Додаток Б

Код сайту

Лістинг Б.1 - Adminbase.php

```
abstract class AdminBase
{
   public static function checkAdmin()
   {       $userId = User::checkLogged();
      $user = User::getUserById($userId);
      if ($user['role'] == 'admin') {
         return true;
      }
      die('Access denied');
   }
Autoload.php
<?php
function__autoload($class_name)
{
   $array_paths = array(
```

```
        '/models/',
        '/components/',
        '/controllers/',
    );
    foreach ($array_paths as $path) {
        $path = ROOT . $path . $class_name . '.php';
        if (is_file($path)) {
            include_once $path;
        }
    }
}
```

Лістинг Б.2 - Cart.php

```
<?php
class Cart
{
    public static function addProduct($id)
    {
        $id = intval($id);
        $productsInCart = array();
        if (isset($_SESSION['products'])) {
            // То заполним наш массив товарами
            $productsInCart = $_SESSION['products'];
        }
        if (array_key_exists($id, $productsInCart)) {
            $productsInCart[$id] ++;
        } else {
```

```
            // Если нет, добавляем id нового товара в корзину с количеством
1
            $productsInCart[$id] = 1;
        }
        $_SESSION['products'] = $productsInCart;

            return self::countItems();
    }
    public static function countItems()
    {       if (isset($_SESSION['products'])) {
            $count = 0;
            foreach ($_SESSION['products'] as $id => $quantity) {
                $count = $count + $quantity;
            }
            return $count;
        } else {
                    return 0;
        }
    }
    public static function getProducts()
    {
        if (isset($_SESSION['products'])) {
            return $_SESSION['products'];
        }
        return false;
    }
    public static function getTotalPrice($products)
    {
        $productsInCart = self::getProducts();
            $total = 0;
```

```
        if ($productsInCart) {
            foreach ($products as $item) {
                $total += $item['price'] * $productsInCart[$item['id']];
            }
        }
        return $total;
    }
    public static function clear()
    {
        if (isset($_SESSION['products'])) {
            unset($_SESSION['products']);
        }
    }
    public static function deleteProduct($id)
    {
        $productsInCart = self::getProducts();
        unset($productsInCart[$id]);
        $_SESSION['products'] = $productsInCart;
    }
}
```

Лістинг Б.3 - Db.php

```
<?php
class Db
{
    public static function getConnection()
    {
        $paramsPath = ROOT . '/config/db_params.php';
        $params = include($paramsPath);
```

```
        $dsn = "mysql:host={$params['host']};dbname={$params['dbname']}";
        $db = new PDO($dsn, $params['user'], $params['password']);
        $db->exec("set names utf8");
        return $db;
    }
}
```

Лістинг Б.4 - Pagination.phhp

```
<?php
class Pagination
{
    private $max = 10;
    private $index = 'page';
    private $current_page;
    private $total;
    private $limit;
    public function__construct($total, $currentPage, $limit, $index)
    {
        $this->total = $total;
        $this->limit = $limit;
        $this->index = $index;
        $this->amount = $this->amount();
        $this->setCurrentPage($currentPage);
    }
    public function get()
    {
        $links = null;
        $limits = $this->limits();
       $html = '<ul class="pagination">';
```

```
            for ($page = $limits[0]; $page <= $limits[1]; $page++) {
                if ($page == $this->current_page) {
                    $links .= '<li class="active"><a href="#">' . $page . '</a></li>';
                } else {
                    # Иначе генерируем ссылку
                    $links .= $this->generateHtml($page);
                }
            }
            if (!is_null($links)) {
                # Если текущая страница не первая
                if ($this->current_page > 1)
                    $links = $this->generateHtml(1, '&lt;') . $links;
                if ($this->current_page < $this->amount)
                    $links .= $this->generateHtml($this->amount, '&gt;');
            }
            $html .= $links . '</ul>';
            return $html;
        }
        private function generateHtml($page, $text = null)
        {
            if (!$text)
                $text = $page;
            $currentURI = rtrim($_SERVER['REQUEST_URI'], '/') . '/';
            $currentURI = preg_replace('~/page-[0-9]+~', '', $currentURI);
            return
                '<li><a href="' . $currentURI . $this->index . $page . '">' . $text .
'</a></li>';
        }
        private function limits()
        {
```

```
            $left = $this->current_page - round($this->max / 2);
            $start = $left > 0 ? $left : 1;
            if ($start + $this->max <= $this->amount) {
                $end = $start > 1 ? $start + $this->max : $this->max;
            } else {
                $end = $this->amount;
                $start = $this->amount - $this->max > 0 ? $this->amount - $this-
>max : 1;
            }
            return
                array($start, $end);
        }
        private function setCurrentPage($currentPage)
        {
            $this->current_page = $currentPage;
            if ($this->current_page > 0) {
                    if ($this->current_page > $this->amount)
                $this->current_page = $this->amount;
            } else
                $this->current_page = 1;
        }
        private function amount()
        {
            return ceil($this->total / $this->limit);
        }
    }
```

Лістинг Б.5 - Routrer.php

```
<?php
```

```
class Router
{
    private $routes;
    public function__construct()
    {
        $routesPath = ROOT . '/config/routes.php';

        $this->routes = include($routesPath);
    }
    private function getURI()
    {
        if (!empty($_SERVER['REQUEST_URI'])) {
            return trim($_SERVER['REQUEST_URI'], '/');
        }
    }
    public function run()
    {
        $uri = $this->getURI();
        foreach ($this->routes as $uriPattern => $path) {
            if (preg_match("~$uriPattern~", $uri)) {
                $internalRoute = preg_replace("~$uriPattern~", $path, $uri);
                $segments = explode('/', $internalRoute);
                $controllerName = array_shift($segments) . 'Controller';
                $controllerName = ucfirst($controllerName);
                $actionName = 'action' . ucfirst(array_shift($segments));
                $parameters = $segments;
                $controllerFile = ROOT . '/controllers/' .
                        $controllerName . '.php';

                if (file_exists($controllerFile)) {
```

```
                    include_once($controllerFile);
                }
                $controllerObject = new $controllerName;
                $result = call_user_func_array(array($controllerObject,
$actionName), $parameters);
                if ($result != null) {
                    break;
                }
            }
        }
    }

}
```

Лістинг Б.6 - Dbparams.php

```
<?php
return array(
    'host' => 'localhost',
    'dbname' => 'store_db',
    'user' => 'root',
    'password' => '',
);
```

Routes.php

```
<?php

return array(
    // Товар:
```

```
        'product/([0-9]+)' => 'product/view/$1', // actionView в ProductController
        // Каталог:
        'catalog' => 'catalog/index', // actionIndex в CatalogController
        // Категория товаров:
        'category/([0-9]+)/page-([0-9]+)' => 'catalog/category/$1/$2', //
actionCategory в CatalogController
        'category/([0-9]+)' => 'catalog/category/$1', // actionCategory в
CatalogController
        // Корзина:
        'cart/checkout' => 'cart/checkout', // actionAdd в CartController
        'cart/delete/([0-9]+)' => 'cart/delete/$1', // actionDelete в CartController
        'cart/add/([0-9]+)' => 'cart/add/$1', // actionAdd в CartController
        'cart/addAjax/([0-9]+)' => 'cart/addAjax/$1', // actionAddAjax в
CartController
        'cart' => 'cart/index', // actionIndex в CartController
        // Пользователь:
        'user/register' => 'user/register',
        'user/login' => 'user/login',
        'user/logout' => 'user/logout',
        'cabinet/edit' => 'cabinet/edit',
        'cabinet' => 'cabinet/index',
        // Управление товарами:
        'admin/product/create' => 'adminProduct/create',
        'admin/product/update/([0-9]+)' => 'adminProduct/update/$1',
        'admin/product/delete/([0-9]+)' => 'adminProduct/delete/$1',
        'admin/product' => 'adminProduct/index',
        // Управление категориями:
        'admin/category/create' => 'adminCategory/create',
        'admin/category/update/([0-9]+)' => 'adminCategory/update/$1',
        'admin/category/delete/([0-9]+)' => 'adminCategory/delete/$1',
```

```
    'admin/category' => 'adminCategory/index',
    // Управление заказами:
    'admin/order/update/([0-9]+)' => 'adminOrder/update/$1',
    'admin/order/delete/([0-9]+)' => 'adminOrder/delete/$1',
    'admin/order/view/([0-9]+)' => 'adminOrder/view/$1',
    'admin/order' => 'adminOrder/index',
    // Админпанель:
    'admin' => 'admin/index',
    // О магазине
    'contacts' => 'site/contact',
    'about' => 'site/about',
    // Главная страница
    'index.php' => 'site/index', // actionIndex в SiteController
    '' => 'site/index', // actionIndex в SiteController
);
```

Лістинг Б.7 - AdminCategoryControlle.php

```
<?php
/**
 * Контроллер AdminCategoryController
 * Управление категориями товаров в админпанели
 */
class AdminCategoryController extends AdminBase
{

    /**
     * Action для страницы "Управление категориями"
     */
    public function actionIndex()
```

```
{
    // Проверка доступа
    self::checkAdmin();

    // Получаем список категорий
    $categoriesList = Category::getCategoriesListAdmin();

    // Подключаем вид
    require_once(ROOT . '/views/admin_category/index.php');
    return true;
}

/**
 * Action для страницы "Добавить категорию"
 */
public function actionCreate()
{
    // Проверка доступа
    self::checkAdmin();

    // Обработка формы
    if (isset($_POST['submit'])) {
        // Если форма отправлена
        // Получаем данные из формы
        $name = $_POST['name'];
        $sortOrder = $_POST['sort_order'];
        $status = $_POST['status'];

        // Флаг ошибок в форме
        $errors = false;
```

```
            // При необходимости можно валидировать значения нужным
образом
            if (!isset($name) || empty($name)) {
                $errors[] = 'Заполните поля';
            }


            if ($errors == false) {
                // Если ошибок нет
                // Добавляем новую категорию
                Category::createCategory($name, $sortOrder, $status);

                // Перенаправляем пользователя на страницу управлениями
категориями
                header("Location: /admin/category");
            }
        }

        require_once(ROOT . '/views/admin_category/create.php');
        return true;
    }

    /**
     * Action для страницы "Редактировать категорию"
     */
    public function actionUpdate($id)
    {
        // Проверка доступа
        self::checkAdmin();
```

```
            // Получаем данные о конкретной категории
            $category = Category::getCategoryById($id);

            // Обработка формы
            if (isset($_POST['submit'])) {
                // Если форма отправлена
                // Получаем данные из формы
                $name = $_POST['name'];
                $sortOrder = $_POST['sort_order'];
                $status = $_POST['status'];

                // Сохраняем изменения
                Category::updateCategoryById($id, $name, $sortOrder, $status);

                // Перенаправляем пользователя на страницу управлениями
категориями
                header("Location: /admin/category");
            }

            // Подключаем вид
            require_once(ROOT . '/views/admin_category/update.php');
            return true;
        }

        /**
         * Action для страницы "Удалить категорию"
         */
        public function actionDelete($id)
        {
```

```
            // Проверка доступа
            self::checkAdmin();

            // Обработка формы
            if (isset($_POST['submit'])) {
                // Если форма отправлена
                // Удаляем категорию
                Category::deleteCategoryById($id);

                // Перенаправляем пользователя на страницу управлениями
товарами
                header("Location: /admin/category");
            }

            // Подключаем вид
            require_once(ROOT . '/views/admin_category/delete.php');
            return true;
        }

    }
```

Лістинг Б.8 - AdminController.php

```
<?php

/**
 * Контроллер AdminController
 * Главная страница в админпанели
 */
class AdminController extends AdminBase
```

```
{
    /**
     * Action для стартовой страницы "Панель администратора"
     */
    public function actionIndex()
    {
        // Проверка доступа
        self::checkAdmin();

        // Подключаем вид
        require_once(ROOT . '/views/admin/index.php');
        return true;
    }

}
```

Лістинг Б.9 - AdminOrderController.php

```
<?php

/**
 * Контроллер AdminOrderController
 * Управление заказами в админпанели
 */
class AdminOrderController extends AdminBase
{

    /**
     * Action для страницы "Управление заказами"
     */
```

```
public function actionIndex()
{
    // Проверка доступа
    self::checkAdmin();

    // Получаем список заказов
    $ordersList = Order::getOrdersList();

    // Подключаем вид
    require_once(ROOT . '/views/admin_order/index.php');
    return true;
}

/**
 * Action для страницы "Редактирование заказа"
 */
public function actionUpdate($id)
{
    // Проверка доступа
    self::checkAdmin();

    // Получаем данные о конкретном заказе
    $order = Order::getOrderById($id);

    // Обработка формы
    if (isset($_POST['submit'])) {
        // Если форма отправлена
        // Получаем данные из формы
        $userName = $_POST['userName'];
        $userPhone = $_POST['userPhone'];
```

```
                $userComment = $_POST['userComment'];
                $date = $_POST['date'];
                $status = $_POST['status'];

                // Сохраняем изменения
                Order::updateOrderById($id, $userName, $userPhone,
$userComment, $date, $status);

                // Перенаправляем пользователя на страницу управлениями
заказами
                header("Location: /admin/order/view/$id");
            }

            // Подключаем вид
            require_once(ROOT . '/views/admin_order/update.php');
            return true;
        }

        /**
         * Action для страницы "Просмотр заказа"
         */
        public function actionView($id)
        {
            // Проверка доступа
            self::checkAdmin();

            // Получаем данные о конкретном заказе
            $order = Order::getOrderById($id);

            // Получаем массив с идентификаторами и количеством товаров
```

```
            $productsQuantity = json_decode($order['products'], true);

            // Получаем массив с индентификаторами товаров
            $productsIds = array_keys($productsQuantity);

            // Получаем список товаров в заказе
            $products = Product::getProdustsByIds($productsIds);

            // Подключаем вид
            require_once(ROOT . '/views/admin_order/view.php');
            return true;
        }

        /**
         * Action для страницы "Удалить заказ"
         */
        public function actionDelete($id)
        {
            // Проверка доступа
            self::checkAdmin();

            // Обработка формы
            if (isset($_POST['submit'])) {
                // Если форма отправлена
                // Удаляем заказ
                Order::deleteOrderById($id);

                // Перенаправляем пользователя на страницу управлениями
товарами
                header("Location: /admin/order");
```

```
        }

        // Подключаем вид
        require_once(ROOT . '/views/admin_order/delete.php');
        return true;
    }

}
```

Лістинг Б.10 - AdminProductController.php

```
<?php

/**
 * Контроллер AdminProductController
 * Управление товарами в админпанели
 */
class AdminProductController extends AdminBase
{

    /**
     * Action для страницы "Управление товарами"
     */
    public function actionIndex()
    {
        // Проверка доступа
        self::checkAdmin();

        // Получаем список товаров
        $productsList = Product::getProductsList();
```

```
    // Подключаем вид
    require_once(ROOT . '/views/admin_product/index.php');
    return true;
}

/**
 * Action для страницы "Добавить товар"
 */
public function actionCreate()
{
    // Проверка доступа
    self::checkAdmin();

    // Получаем список категорий для выпадающего списка
    $categoriesList = Category::getCategoriesListAdmin();

    // Обработка формы
    if (isset($_POST['submit'])) {
        // Если форма отправлена
        // Получаем данные из формы
        $options['name'] = $_POST['name'];
        $options['code'] = $_POST['code'];
        $options['price'] = $_POST['price'];
        $options['category_id'] = $_POST['category_id'];
        $options['brand'] = $_POST['brand'];
        $options['availability'] = $_POST['availability'];
        $options['description'] = $_POST['description'];
        $options['is_new'] = $_POST['is_new'];
        $options['is_recommended'] = $_POST['is_recommended'];
```

```
            $options['status'] = $_POST['status'];

            // Флаг ошибок в форме
            $errors = false;

            // При необходимости можно валидировать значения нужным
образом
            if (!isset($options['name']) || empty($options['name'])) {
                $errors[] = 'Заполните поля';
            }

            if ($errors == false) {
                // Если ошибок нет
                // Добавляем новый товар
                $id = Product::createProduct($options);

                // Если запись добавлена
                if ($id) {
                    // Проверим, загружалось ли через форму изображение
                    if (is_uploaded_file($_FILES["image"]["tmp_name"])) {
                        // Если загружалось, переместим его в нужную папке,
дадим новое имя
                        move_uploaded_file($_FILES["image"]["tmp_name"],
$_SERVER['DOCUMENT_ROOT'] . "/upload/images/products/{$id}.jpg");
                    }
                };

                // Перенаправляем пользователя на страницу управлениями
товарами
                header("Location: /admin/product");
```

```
            }
        }

        // Подключаем вид
        require_once(ROOT . '/views/admin_product/create.php');
        return true;
    }

    /**
     * Action для страницы "Редактировать товар"
     */
    public function actionUpdate($id)
    {
        // Проверка доступа
        self::checkAdmin();

        // Получаем список категорий для выпадающего списка
        $categoriesList = Category::getCategoriesListAdmin();

        // Получаем данные о конкретном заказе
        $product = Product::getProductById($id);

        // Обработка формы
        if (isset($_POST['submit'])) {
            // Если форма отправлена
            // Получаем данные из формы редактирования. При
необходимости можно валидировать значения
            $options['name'] = $_POST['name'];
            $options['code'] = $_POST['code'];
            $options['price'] = $_POST['price'];
```

```
                $options['category_id'] = $_POST['category_id'];
                $options['brand'] = $_POST['brand'];
                $options['availability'] = $_POST['availability'];
                $options['description'] = $_POST['description'];
                $options['is_new'] = $_POST['is_new'];
                $options['is_recommended'] = $_POST['is_recommended'];
                $options['status'] = $_POST['status'];

                // Сохраняем изменения
                if (Product::updateProductById($id, $options)) {



                    // Если запись сохранена
                    // Проверим, загружалось ли через форму изображение
                    if (is_uploaded_file($_FILES["image"]["tmp_name"])) {

                        // Если загружалось, переместим его в нужную папке,
дадим новое имя
                        move_uploaded_file($_FILES["image"]["tmp_name"],
$_SERVER['DOCUMENT_ROOT'] . "/upload/images/products/{$id}.jpg");
                    }
                }

                // Перенаправляем пользователя на страницу управлениями
товарами
                header("Location: /admin/product");
            }

            // Подключаем вид
            require_once(ROOT . '/views/admin_product/update.php');
```

```
        return true;
    }

    /**
     * Action для страницы "Удалить товар"
     */
    public function actionDelete($id)
    {
        // Проверка доступа
        self::checkAdmin();

        // Обработка формы
        if (isset($_POST['submit'])) {
            // Если форма отправлена
            // Удаляем товар
            Product::deleteProductById($id);

            // Перенаправляем пользователя на страницу управлениями товарами
            header("Location: /admin/product");
        }

        // Подключаем вид
        require_once(ROOT . '/views/admin_product/delete.php');
        return true;
    }

}
```

Лістинг Б.11 - CabinetController.php

```
<?php

/**
 * Контроллер CabinetController
 * Кабинет пользователя
 */
class CabinetController
{

    /**
     * Action для страницы "Кабинет пользователя"
     */
    public function actionIndex()
    {
        // Получаем идентификатор пользователя из сессии
        $userId = User::checkLogged();

        // Получаем информацию о пользователе из БД
        $user = User::getUserById($userId);

        // Подключаем вид
        require_once(ROOT . '/views/cabinet/index.php');
        return true;
    }

    /**
     * Action для страницы "Редактирование данных пользователя"
     */
```

```
public function actionEdit()
{
    // Получаем идентификатор пользователя из сессии
    $userId = User::checkLogged();

    // Получаем информацию о пользователе из БД
    $user = User::getUserById($userId);

    // Заполняем переменные для полей формы
    $name = $user['name'];
    $password = $user['password'];

    // Флаг результата
    $result = false;

    // Обработка формы
    if (isset($_POST['submit'])) {
        // Если форма отправлена
        // Получаем данные из формы редактирования
        $name = $_POST['name'];
        $password = $_POST['password'];

        // Флаг ошибок
        $errors = false;

        // Валидируем значения
        if (!User::checkName($name)) {
            $errors[] = 'Имя не должно быть короче 2-х символов';
        }
        if (!User::checkPassword($password)) {
```

```
                $errors[] = 'Пароль не должен быть короче 6-ти символов';
            }

            if ($errors == false) {
                // Если ошибок нет, сохраняет изменения профиля
                $result = User::edit($userId, $name, $password);
            }
        }

        // Подключаем вид
        require_once(ROOT . '/views/cabinet/edit.php');
        return true;
    }

}
```

Лістинг Б.12 - CartController.php

```
<?php

/**
 * Контроллер CartController
 * Корзина
 */
class CartController
{

    /**
     * Action для добавления товара в корзину синхронным
запросом<br/>
```

```
     * (для примера, не используется)
     * @param integer $id <p>id товара</p>
     */
    public function actionAdd($id)
    {
        // Добавляем товар в корзину
        Cart::addProduct($id);

        // Возвращаем пользователя на страницу с которой он пришел
        $referrer = $_SERVER['HTTP_REFERER'];
        header("Location: $referrer");
    }

    /**
     * Action для добавления товара в корзину при помощи асинхронного запроса (ajax)
     * @param integer $id <p>id товара</p>
     */
    public function actionAddAjax($id)
    {
        // Добавляем товар в корзину и печатаем результат: количество товаров в корзине
        echo Cart::addProduct($id);
        return true;
    }

    /**
     * Action для добавления товара в корзину синхронным запросом
     * @param integer $id <p>id товара</p>
     */
```

```
    public function actionDelete($id)
    {
        // Удаляем заданный товар из корзины
        Cart::deleteProduct($id);

        // Возвращаем пользователя в корзину
        header("Location: /cart");
    }

    /**
     * Action для страницы "Корзина"
     */
    public function actionIndex()
    {
        // Список категорий для левого меню
        $categories = Category::getCategoriesList();

        // Получим идентификаторы и количество товаров в корзине
        $productsInCart = Cart::getProducts();

        if ($productsInCart) {
            // Если в корзине есть товары, получаем полную информацию о
товарах для списка
            // Получаем массив только с идентификаторами товаров
            $productsIds = array_keys($productsInCart);

            // Получаем массив с полной информацией о необходимых
товарах
            $products = Product::getProdustsByIds($productsIds);
```

```
            // Получаем общую стоимость товаров
            $totalPrice = Cart::getTotalPrice($products);
        }

        // Подключаем вид
        require_once(ROOT . '/views/cart/index.php');
        return true;
    }

    /**
     * Action для страницы "Оформление покупки"
     */
    public function actionCheckout()
    {
        // Получием данные из корзины
        $productsInCart = Cart::getProducts();

        // Если товаров нет, отправляем пользователи искать товары на
главную
        if ($productsInCart == false) {
            header("Location: /");
        }

        // Список категорий для левого меню
        $categories = Category::getCategoriesList();

        // Находим общую стоимость
        $productsIds = array_keys($productsInCart);
        $products = Product::getProdustsByIds($productsIds);
        $totalPrice = Cart::getTotalPrice($products);
```

```
// Количество товаров
$totalQuantity = Cart::countItems();

// Поля для формы
$userName = false;
$userPhone = false;
$userComment = false;

// Статус успешного оформления заказа
$result = false;

// Проверяем является ли пользователь гостем
if (!User::isGuest()) {
    // Если пользователь не гость
    // Получаем информацию о пользователе из БД
    $userId = User::checkLogged();
    $user = User::getUserById($userId);
    $userName = $user['name'];
} else {
    // Если гость, поля формы останутся пустыми
    $userId = false;
}

// Обработка формы
if (isset($_POST['submit'])) {
    // Если форма отправлена
    // Получаем данные из формы
    $userName = $_POST['userName'];
    $userPhone = $_POST['userPhone'];
```

```
            $userComment = $_POST['userComment'];

            // Флаг ошибок
            $errors = false;

            // Валидация полей
            if (!User::checkName($userName)) {
                $errors[] = 'Неправильное имя';
            }
            if (!User::checkPhone($userPhone)) {
                $errors[] = 'Неправильный телефон';
            }


            if ($errors == false) {
                // Если ошибок нет
                // Сохраняем заказ в базе данных
                $result = Order::save($userName, $userPhone, $userComment,
$userId, $productsInCart);

                if ($result) {
                    // Если заказ успешно сохранен
                    // Оповещаем администратора о новом заказе по почте
                    $adminEmail = 'php.start@mail.ru';
                    $message = '<a href="http://digital-
mafia.net/admin/orders">Список заказов</a>';
                    $subject = 'Новый заказ!';
                    mail($adminEmail, $subject, $message);

                    // Очищаем корзину
```

```
                Cart::clear();
            }
        }
    }

    // Подключаем вид
    require_once(ROOT . '/views/cart/checkout.php');
    return true;
  }

}
```

Лістинг Б.13 - CatalogController.php

```
<?php

/**
 * Контроллер CatalogController
 * Каталог товаров
 */
class CatalogController
{

  /**
   * Action для страницы "Каталог товаров"
   */
  public function actionIndex()
  {
    // Список категорий для левого меню
    $categories = Category::getCategoriesList();
```

```
        // Список последних товаров
        $latestProducts = Product::getLatestProducts(12);

        // Подключаем вид
        require_once(ROOT . '/views/catalog/index.php');
        return true;
    }

    /**
     * Action для страницы "Категория товаров"
     */
    public function actionCategory($categoryId, $page = 1)
    {
        // Список категорий для левого меню
        $categories = Category::getCategoriesList();

        // Список товаров в категории
        $categoryProducts = Product::getProductsListByCategory($categoryId,
$page);

        // Общее количетсво товаров (необходимо для постраничной
навигации)
        $total = Product::getTotalProductsInCategory($categoryId);

        // Создаем объект Pagination - постраничная навигация
        $pagination = new Pagination($total, $page,
Product::SHOW_BY_DEFAULT, 'page-');

        // Подключаем вид
        require_once(ROOT . '/views/catalog/category.php');
```

```
        return true;
    }
```

Лістинг Б.14 - ProductController.php

```
<?php

/**
 * Контроллер ProductController
 * Товар
 */
class ProductController
{

    /**
     * Action для страницы просмотра товара
     * @param integer $productId <p>id товара</p>
     */
    public function actionView($productId)
    {
        // Список категорий для левого меню
        $categories = Category::getCategoriesList();

        // Получаем инфомрацию о товаре
        $product = Product::getProductById($productId);

        // Подключаем вид
        require_once(ROOT . '/views/product/view.php');
        return true;
    }
```

```
}
```

Лістинг Б.15 - SiteController.php

```
<?php

/**
 * Контроллер CartController
 */
class SiteController
{

    /**
     * Action для главной страницы
     */
    public function actionIndex()
    {
        // Список категорий для левого меню
        $categories = Category::getCategoriesList();

        // Список последних товаров
        $latestProducts = Product::getLatestProducts(6);

        // Список товаров для слайдера
        $sliderProducts = Product::getRecommendedProducts();

        // Подключаем вид
        require_once(ROOT . '/views/site/index.php');
        return true;
```

```
}

/**
 * Action для страницы "Контакты"
 */
public function actionContact()
{

    // Переменные для формы
    $userEmail = false;
    $userText = false;
    $result = false;

    // Обработка формы
    if (isset($_POST['submit'])) {
        // Если форма отправлена
        // Получаем данные из формы
        $userEmail = $_POST['userEmail'];
        $userText = $_POST['userText'];

        // Флаг ошибок
        $errors = false;

        // Валидация полей
        if (!User::checkEmail($userEmail)) {
            $errors[] = 'Неправильный email';
        }

        if ($errors == false) {
            // Если ошибок нет
```

```
                // Отправляем письмо администратору
                $adminEmail = 'php.start@mail.ru';
                $message = "Текст: {$userText}. От {$userEmail}";
                $subject = 'Тема письма';
                $result = mail($adminEmail, $subject, $message);
                $result = true;
            }
        }

        // Подключаем вид
        require_once(ROOT . '/views/site/contact.php');
        return true;
    }

    /**
     * Action для страницы "О магазине"
     */
    public function actionAbout()
    {
        // Подключаем вид
        require_once(ROOT . '/views/site/about.php');
        return true;
    }

}
```

Лістинг Б.16 - UserController.php

```
<?php
```

```
/**
 * Контроллер UserController
 */
class UserController
{
    /**
     * Action для страницы "Регистрация"
     */
    public function actionRegister()
    {
        // Переменные для формы
        $name = false;
        $email = false;
        $password = false;
        $result = false;

        // Обработка формы
        if (isset($_POST['submit'])) {
            // Если форма отправлена
            // Получаем данные из формы
            $name = $_POST['name'];
            $email = $_POST['email'];
            $password = $_POST['password'];

            // Флаг ошибок
            $errors = false;

            // Валидация полей
            if (!User::checkName($name)) {
                $errors[] = 'Имя не должно быть короче 2-х символов';
```

```
        }
        if (!User::checkEmail($email)) {
            $errors[] = 'Неправильный email';
        }
        if (!User::checkPassword($password)) {
            $errors[] = 'Пароль не должен быть короче 6-ти символов';
        }
        if (User::checkEmailExists($email)) {
            $errors[] = 'Такой email уже используется';
        }

        if ($errors == false) {
            // Если ошибок нет
            // Регистрируем пользователя
            $result = User::register($name, $email, $password);
        }
    }

    // Подключаем вид
    require_once(ROOT . '/views/user/register.php');
    return true;
}

/**
 * Action для страницы "Вход на сайт"
 */
public function actionLogin()
{
    // Переменные для формы
    $email = false;
```

```
$password = false;

// Обработка формы
if (isset($_POST['submit'])) {
    // Если форма отправлена
    // Получаем данные из формы
    $email = $_POST['email'];
    $password = $_POST['password'];

    // Флаг ошибок
    $errors = false;

    // Валидация полей
    if (!User::checkEmail($email)) {
        $errors[] = 'Неправильный email';
    }
    if (!User::checkPassword($password)) {
        $errors[] = 'Пароль не должен быть короче 6-ти символов';
    }

    // Проверяем существует ли пользователь
    $userId = User::checkUserData($email, $password);

    if ($userId == false) {
        // Если данные неправильные - показываем ошибку
        $errors[] = 'Неправильные данные для входа на сайт';
    } else {
        // Если данные правильные, запоминаем пользователя (сессия)
        User::auth($userId);
```

```
                // Перенаправляем пользователя в закрытую часть - кабинет
                header("Location: /cabinet");
            }
        }

        // Подключаем вид
        require_once(ROOT . '/views/user/login.php');
        return true;
    }

    /**
     * Удаляем данные о пользователе из сессии
     */
    public function actionLogout()
    {
        // Стартуем сессию
        session_start();

        // Удаляем информацию о пользователе из сессии
        unset($_SESSION["user"]);

        // Перенаправляем пользователя на главную страницу
        header("Location: /");
    }

}
```

Лістинг Б.17 - Category.php

```
<?php
```

```
/**
 * Класс Category - модель для работы с категориями товаров
 */
class Category
{

    /**
     * Возвращает массив категорий для списка на сайте
     * @return array <p>Массив с категориями</p>
     */
    public static function getCategoriesList()
    {
        // Соединение с БД
        $db = Db::getConnection();

        // Запрос к БД
        $result = $db->query('SELECT id, name FROM category WHERE
status = "1" ORDER BY sort_order, name ASC');

        // Получение и возврат результатов
        $i = 0;
        $categoryList = array();
        while ($row = $result->fetch()) {
            $categoryList[$i]['id'] = $row['id'];
            $categoryList[$i]['name'] = $row['name'];
            $i++;
        }
        return $categoryList;
    }
```

```
        /**
         * Возвращает массив категорий для списка в админпанели <br/>
         * (при этом в результат попадают и включенные и выключенные
категории)
         * @return array <p>Массив категорий</p>
         */
        public static function getCategoriesListAdmin()
        {
            // Соединение с БД
            $db = Db::getConnection();

            // Запрос к БД
            $result = $db->query('SELECT id, name, sort_order, status FROM
category ORDER BY sort_order ASC');

            // Получение и возврат результатов
            $categoryList = array();
            $i = 0;
            while ($row = $result->fetch()) {
                $categoryList[$i]['id'] = $row['id'];
                $categoryList[$i]['name'] = $row['name'];
                $categoryList[$i]['sort_order'] = $row['sort_order'];
                $categoryList[$i]['status'] = $row['status'];
                $i++;
            }
            return $categoryList;
        }

        /**
```

```
     * Удаляет категорию с заданным id
     * @param integer $id
     * @return boolean <p>Результат выполнения метода</p>
     */
    public static function deleteCategoryById($id)
    {
        // Соединение с БД
        $db = Db::getConnection();

        // Текст запроса к БД
        $sql = 'DELETE FROM category WHERE id = :id';

        // Получение и возврат результатов. Используется подготовленный
запрос
        $result = $db->prepare($sql);
        $result->bindParam(':id', $id, PDO::PARAM_INT);
        return $result->execute();
    }

    /**
     * Редактирование категории с заданным id
     * @param integer $id <p>id категории</p>
     * @param string $name <p>Название</p>
     * @param integer $sortOrder <p>Порядковый номер</p>
     * @param integer $status <p>Статус <i>(включено "1", выключено
"0")</i></p>
     * @return boolean <p>Результат выполнения метода</p>
     */
    public static function updateCategoryById($id, $name, $sortOrder,
$status)
```

```
        {
            // Соединение с БД
            $db = Db::getConnection();

            // Текст запроса к БД
            $sql = "UPDATE category
                SET
                    name = :name,
                    sort_order = :sort_order,
                    status = :status
                WHERE id = :id";

            // Получение и возврат результатов. Используется подготовленный
запрос
            $result = $db->prepare($sql);
            $result->bindParam(':id', $id, PDO::PARAM_INT);
            $result->bindParam(':name', $name, PDO::PARAM_STR);
            $result->bindParam(':sort_order', $sortOrder, PDO::PARAM_INT);
            $result->bindParam(':status', $status, PDO::PARAM_INT);
            return $result->execute();
        }

        /**
         * Возвращает категорию с указанным id
         * @param integer $id <p>id категории</p>
         * @return array <p>Массив с информацией о категории</p>
         */
        public static function getCategoryById($id)
        {
            // Соединение с БД
```

```
    $db = Db::getConnection();

    // Текст запроса к БД
    $sql = 'SELECT * FROM category WHERE id = :id';

    // Используется подготовленный запрос
    $result = $db->prepare($sql);
    $result->bindParam(':id', $id, PDO::PARAM_INT);

    // Указываем, что хотим получить данные в виде массива
    $result->setFetchMode(PDO::FETCH_ASSOC);

    // Выполняем запрос
    $result->execute();

    // Возвращаем данные
    return $result->fetch();
}

/**
 * Возвращает текстое пояснение статуса для категории :<br/>
 * <i>0 - Скрыта, 1 - Отображается</i>
 * @param integer $status <p>Статус</p>
 * @return string <p>Текстовое пояснение</p>
 */
public static function getStatusText($status)
{
    switch ($status) {
        case '1':
            return 'Отображается';
```

```
                break;
            case '0':
                return 'Скрыта';
                break;
        }
    }

    /**
     * Добавляет новую категорию
     * @param string $name <p>Название</p>
     * @param integer $sortOrder <p>Порядковый номер</p>
     * @param integer $status <p>Статус <i>(включено "1", выключено "0")</i></p>
     * @return boolean <p>Результат добавления записи в таблицу</p>
     */
    public static function createCategory($name, $sortOrder, $status)
    {
        // Соединение с БД
        $db = Db::getConnection();

        // Текст запроса к БД
        $sql = 'INSERT INTO category (name, sort_order, status) '
            . 'VALUES (:name, :sort_order, :status)';

        // Получение и возврат результатов. Используется подготовленный запрос
        $result = $db->prepare($sql);
        $result->bindParam(':name', $name, PDO::PARAM_STR);
        $result->bindParam(':sort_order', $sortOrder, PDO::PARAM_INT);
        $result->bindParam(':status', $status, PDO::PARAM_INT);
```

```
            return $result->execute();
        }

    }
```

Лістинг Б.18 - Order.php

```
<?php

/**
 * Класс Order - модель для работы с заказами
 */
class Order
{

    /**
     * Сохранение заказа
     * @param string $userName <p>Имя</p>
     * @param string $userPhone <p>Телефон</p>
     * @param string $userComment <p>Комментарий</p>
     * @param integer $userId <p>id пользователя</p>
     * @param array $products <p>Массив с товарами</p>
     * @return boolean <p>Результат выполнения метода</p>
     */
    public static function save($userName, $userPhone, $userComment,
$userId, $products)
    {
        // Соединение с БД
        $db = Db::getConnection();
```

```
            // Текст запроса к БД
            $sql = 'INSERT INTO product_order (user_name, user_phone,
user_comment, user_id, products) '
                    . 'VALUES (:user_name, :user_phone, :user_comment, :user_id,
:products)';

            $products = json_encode($products);

            $result = $db->prepare($sql);
            $result->bindParam(':user_name', $userName, PDO::PARAM_STR);
            $result->bindParam(':user_phone', $userPhone, PDO::PARAM_STR);
            $result->bindParam(':user_comment', $userComment,
PDO::PARAM_STR);
            $result->bindParam(':user_id', $userId, PDO::PARAM_STR);
            $result->bindParam(':products', $products, PDO::PARAM_STR);

            return $result->execute();
        }

        /**
         * Возвращает список заказов
         * @return array <p>Список заказов</p>
         */
        public static function getOrdersList()
        {
            // Соединение с БД
            $db = Db::getConnection();

            // Получение и возврат результатов
```

```
        $result = $db->query('SELECT id, user_name, user_phone, date, status
FROM product_order ORDER BY id DESC');
        $ordersList = array();
        $i = 0;
        while ($row = $result->fetch()) {
            $ordersList[$i]['id'] = $row['id'];
            $ordersList[$i]['user_name'] = $row['user_name'];
            $ordersList[$i]['user_phone'] = $row['user_phone'];
            $ordersList[$i]['date'] = $row['date'];
            $ordersList[$i]['status'] = $row['status'];
            $i++;
        }
        return $ordersList;
    }

    /**
     * Возвращает текстое пояснение статуса для заказа :<br/>
     * <i>1 - Новый заказ, 2 - В обработке, 3 - Доставляется, 4 -
Закрыт</i>
     * @param integer $status <p>Статус</p>
     * @return string <p>Текстовое пояснение</p>
     */
    public static function getStatusText($status)
    {
        switch ($status) {
            case '1':
                return 'Новый заказ';
                break;
            case '2':
                return 'В обработке';
```

```
            break;
        case '3':
            return 'Доставляется';
            break;
        case '4':
            return 'Закрыт';
            break;
    }
}

/**
 * Возвращает заказ с указанным id
 * @param integer $id <p>id</p>
 * @return array <p>Массив с информацией о заказе</p>
 */
public static function getOrderById($id)
{
    // Соединение с БД
    $db = Db::getConnection();

    // Текст запроса к БД
    $sql = 'SELECT * FROM product_order WHERE id = :id';

    $result = $db->prepare($sql);
    $result->bindParam(':id', $id, PDO::PARAM_INT);

    // Указываем, что хотим получить данные в виде массива
    $result->setFetchMode(PDO::FETCH_ASSOC);

    // Выполняем запрос
```

```
        $result->execute();

        // Возвращаем данные
        return $result->fetch();
    }

    /**
     * Удаляет заказ с заданным id
     * @param integer $id <p>id заказа</p>
     * @return boolean <p>Результат выполнения метода</p>
     */
    public static function deleteOrderById($id)
    {
        // Соединение с БД
        $db = Db::getConnection();

        // Текст запроса к БД
        $sql = 'DELETE FROM product_order WHERE id = :id';

        // Получение и возврат результатов. Используется подготовленный
запрос
        $result = $db->prepare($sql);
        $result->bindParam(':id', $id, PDO::PARAM_INT);
        return $result->execute();
    }

    /**
     * Редактирует заказ с заданным id
     * @param integer $id <p>id товара</p>
     * @param string $userName <p>Имя клиента</p>
```

```
     * @param string $userPhone <p>Телефон клиента</p>
     * @param string $userComment <p>Комментарий клиента</p>
     * @param string $date <p>Дата оформления</p>
     * @param integer $status <p>Статус <i>(включено "1", выключено "0")</i></p>
     * @return boolean <p>Результат выполнения метода</p>
     */
    public static function updateOrderById($id, $userName, $userPhone, $userComment, $date, $status)
    {
        // Соединение с БД
        $db = Db::getConnection();

        // Текст запроса к БД
        $sql = "UPDATE product_order
            SET
                user_name = :user_name,
                user_phone = :user_phone,
                user_comment = :user_comment,
                date = :date,
                status = :status
            WHERE id = :id";

        // Получение и возврат результатов. Используется подготовленный запрос
        $result = $db->prepare($sql);
        $result->bindParam(':id', $id, PDO::PARAM_INT);
        $result->bindParam(':user_name', $userName, PDO::PARAM_STR);
        $result->bindParam(':user_phone', $userPhone, PDO::PARAM_STR);
```

```
            $result->bindParam(':user_comment', $userComment,
PDO::PARAM_STR);
            $result->bindParam(':date', $date, PDO::PARAM_STR);
            $result->bindParam(':status', $status, PDO::PARAM_INT);
            return $result->execute();
        }

    }
```

Лістинг Б.19 - Product.php

```
    <?php

    /**
     * Класс Product - модель для работы с товарами
     */
    class Product
    {

        // Количество отображаемых товаров по умолчанию
        const SHOW_BY_DEFAULT = 6;

        /**
         * Возвращает массив последних товаров
         * @param type $count [optional] <p>Количество</p>
         * @param type $page [optional] <p>Номер текущей страницы</p>
         * @return array <p>Массив с товарами</p>
         */
        public static function getLatestProducts($count =
self::SHOW_BY_DEFAULT)
```

```
{
    // Соединение с БД
    $db = Db::getConnection();

    // Текст запроса к БД
    $sql = 'SELECT id, name, price, is_new FROM product '
            . 'WHERE status = "1" ORDER BY id DESC '
            . 'LIMIT :count';

    // Используется подготовленный запрос
    $result = $db->prepare($sql);
    $result->bindParam(':count', $count, PDO::PARAM_INT);

    // Указываем, что хотим получить данные в виде массива
    $result->setFetchMode(PDO::FETCH_ASSOC);

    // Выполнение коменды
    $result->execute();

    // Получение и возврат результатов
    $i = 0;
    $productsList = array();
    while ($row = $result->fetch()) {
        $productsList[$i]['id'] = $row['id'];
        $productsList[$i]['name'] = $row['name'];
        $productsList[$i]['price'] = $row['price'];
        $productsList[$i]['is_new'] = $row['is_new'];
        $i++;
    }
    return $productsList;
```

```
}

/**
 * Возвращает список товаров в указанной категории
 * @param type $categoryId <p>id категории</p>
 * @param type $page [optional] <p>Номер страницы</p>
 * @return type <p>Массив с товарами</p>
 */
public static function getProductsListByCategory($categoryId, $page = 1)
{
    $limit = Product::SHOW_BY_DEFAULT;
    // Смещение (для запроса)
    $offset = ($page - 1) * self::SHOW_BY_DEFAULT;

    // Соединение с БД
    $db = Db::getConnection();

    // Текст запроса к БД
    $sql = 'SELECT id, name, price, is_new FROM product '
            . 'WHERE status = 1 AND category_id = :category_id '
            . 'ORDER BY id ASC LIMIT :limit OFFSET :offset';

    // Используется подготовленный запрос
    $result = $db->prepare($sql);
    $result->bindParam(':category_id', $categoryId, PDO::PARAM_INT);
    $result->bindParam(':limit', $limit, PDO::PARAM_INT);
    $result->bindParam(':offset', $offset, PDO::PARAM_INT);

    // Выполнение коменды
    $result->execute();
```

```
    // Получение и возврат результатов
    $i = 0;
    $products = array();
    while ($row = $result->fetch()) {
        $products[$i]['id'] = $row['id'];
        $products[$i]['name'] = $row['name'];
        $products[$i]['price'] = $row['price'];
        $products[$i]['is_new'] = $row['is_new'];
        $i++;
    }
    return $products;
}

/**
 * Возвращает продукт с указанным id
 * @param integer $id <p>id товара</p>
 * @return array <p>Массив с информацией о товаре</p>
 */
public static function getProductById($id)
{
    // Соединение с БД
    $db = Db::getConnection();

    // Текст запроса к БД
    $sql = 'SELECT * FROM product WHERE id = :id';

    // Используется подготовленный запрос
    $result = $db->prepare($sql);
    $result->bindParam(':id', $id, PDO::PARAM_INT);
```

```
        // Указываем, что хотим получить данные в виде массива
        $result->setFetchMode(PDO::FETCH_ASSOC);

        // Выполнение коменды
        $result->execute();

        // Получение и возврат результатов
        return $result->fetch();
    }

    /**
     * Возвращаем количество товаров в указанной категории
     * @param integer $categoryId
     * @return integer
     */
    public static function getTotalProductsInCategory($categoryId)
    {
        // Соединение с БД
        $db = Db::getConnection();

        // Текст запроса к БД
        $sql = 'SELECT count(id) AS count FROM product WHERE
status="1" AND category_id = :category_id';

        // Используется подготовленный запрос
        $result = $db->prepare($sql);
        $result->bindParam(':category_id', $categoryId, PDO::PARAM_INT);

        // Выполнение коменды
```

```
        $result->execute();

        // Возвращаем значение count - количество
        $row = $result->fetch();
        return $row['count'];
    }

    /**
     * Возвращает список товаров с указанными индентификторами
     * @param array $idsArray <p>Массив с идентификаторами</p>
     * @return array <p>Массив со списком товаров</p>
     */
    public static function getProdustsByIds($idsArray)
    {
        // Соединение с БД
        $db = Db::getConnection();

        // Превращаем массив в строку для формирования условия в
запросе
        $idsString = implode(',', $idsArray);

        // Текст запроса к БД
        $sql = "SELECT * FROM product WHERE status='1' AND id IN
($idsString)";

        $result = $db->query($sql);

        // Указываем, что хотим получить данные в виде массива
        $result->setFetchMode(PDO::FETCH_ASSOC);
```

```
        // Получение и возврат результатов
        $i = 0;
        $products = array();
        while ($row = $result->fetch()) {
            $products[$i]['id'] = $row['id'];
            $products[$i]['code'] = $row['code'];
            $products[$i]['name'] = $row['name'];
            $products[$i]['price'] = $row['price'];
            $i++;
        }
        return $products;
    }

    /**
     * Возвращает список рекомендуемых товаров
     * @return array <p>Массив с товарами</p>
     */
    public static function getRecommendedProducts()
    {
        // Соединение с БД
        $db = Db::getConnection();

        // Получение и возврат результатов
        $result = $db->query('SELECT id, name, price, is_new FROM product
'
                . 'WHERE status = "1" AND is_recommended = "1" '
                . 'ORDER BY id DESC');
        $i = 0;
        $productsList = array();
        while ($row = $result->fetch()) {
```

```
            $productsList[$i]['id'] = $row['id'];
            $productsList[$i]['name'] = $row['name'];
            $productsList[$i]['price'] = $row['price'];
            $productsList[$i]['is_new'] = $row['is_new'];
            $i++;
        }
        return $productsList;
    }

    /**
     * Возвращает список товаров
     * @return array <p>Массив с товарами</p>
     */
    public static function getProductsList()
    {
        // Соединение с БД
        $db = Db::getConnection();

        // Получение и возврат результатов
        $result = $db->query('SELECT id, name, price, code FROM product
ORDER BY id ASC');
        $productsList = array();
        $i = 0;
        while ($row = $result->fetch()) {
            $productsList[$i]['id'] = $row['id'];
            $productsList[$i]['name'] = $row['name'];
            $productsList[$i]['code'] = $row['code'];
            $productsList[$i]['price'] = $row['price'];
            $i++;
        }
```

```
        return $productsList;
    }

    /**
     * Удаляет товар с указанным id
     * @param integer $id <p>id товара</p>
     * @return boolean <p>Результат выполнения метода</p>
     */
    public static function deleteProductById($id)
    {
        // Соединение с БД
        $db = Db::getConnection();

        // Текст запроса к БД
        $sql = 'DELETE FROM product WHERE id = :id';

        // Получение и возврат результатов. Используется подготовленный
запрос
        $result = $db->prepare($sql);
        $result->bindParam(':id', $id, PDO::PARAM_INT);
        return $result->execute();
    }

    /**
     * Редактирует товар с заданным id
     * @param integer $id <p>id товара</p>
     * @param array $options <p>Массив с информацей о товаре</p>
     * @return boolean <p>Результат выполнения метода</p>
     */
    public static function updateProductById($id, $options)
```

```
        {
            // Соединение с БД
            $db = Db::getConnection();

            // Текст запроса к БД
            $sql = "UPDATE product
                SET
                    name = :name,
                    code = :code,
                    price = :price,
                    category_id = :category_id,
                    brand = :brand,
                    availability = :availability,
                    description = :description,
                    is_new = :is_new,
                    is_recommended = :is_recommended,
                    status = :status
                WHERE id = :id";

            // Получение и возврат результатов. Используется подготовленный
запрос
            $result = $db->prepare($sql);
            $result->bindParam(':id', $id, PDO::PARAM_INT);
            $result->bindParam(':name', $options['name'], PDO::PARAM_STR);
            $result->bindParam(':code', $options['code'], PDO::PARAM_STR);
            $result->bindParam(':price', $options['price'], PDO::PARAM_STR);
            $result->bindParam(':category_id', $options['category_id'],
PDO::PARAM_INT);
            $result->bindParam(':brand', $options['brand'], PDO::PARAM_STR);
```

```
            $result->bindParam(':availability', $options['availability'],
PDO::PARAM_INT);
            $result->bindParam(':description', $options['description'],
PDO::PARAM_STR);
            $result->bindParam(':is_new', $options['is_new'],
PDO::PARAM_INT);
            $result->bindParam(':is_recommended', $options['is_recommended'],
PDO::PARAM_INT);
            $result->bindParam(':status', $options['status'], PDO::PARAM_INT);
            return $result->execute();
        }

        /**
         * Добавляет новый товар
         * @param array $options <p>Массив с информацией о товаре</p>
         * @return integer <p>id добавленной в таблицу записи</p>
         */
        public static function createProduct($options)
        {
            // Соединение с БД
            $db = Db::getConnection();

            // Текст запроса к БД
            $sql = 'INSERT INTO product '
                . '(name, code, price, category_id, brand, availability,'
                . 'description, is_new, is_recommended, status)'
                . 'VALUES '
                . '(:name, :code, :price, :category_id, :brand, :availability,'
                . ':description, :is_new, :is_recommended, :status)';
```

```
            // Получение и возврат результатов. Используется подготовленный
запрос
            $result = $db->prepare($sql);
            $result->bindParam(':name', $options['name'], PDO::PARAM_STR);
            $result->bindParam(':code', $options['code'], PDO::PARAM_STR);
            $result->bindParam(':price', $options['price'], PDO::PARAM_STR);
            $result->bindParam(':category_id', $options['category_id'],
PDO::PARAM_INT);
            $result->bindParam(':brand', $options['brand'], PDO::PARAM_STR);
            $result->bindParam(':availability', $options['availability'],
PDO::PARAM_INT);
            $result->bindParam(':description', $options['description'],
PDO::PARAM_STR);
            $result->bindParam(':is_new', $options['is_new'],
PDO::PARAM_INT);
            $result->bindParam(':is_recommended', $options['is_recommended'],
PDO::PARAM_INT);
            $result->bindParam(':status', $options['status'], PDO::PARAM_INT);
            if ($result->execute()) {
                // Если запрос выполенен успешно, возвращаем id добавленной
записи
                return $db->lastInsertId();
            }
            // Иначе возвращаем 0
            return 0;
        }

        /**
         * Возвращает текстое пояснение наличия товара:<br/>
         * <i>0 - Под заказ, 1 - В наличии</i>
```

```
     * @param integer $availability <p>Статус</p>
     * @return string <p>Текстовое пояснение</p>
     */
    public static function getAvailabilityText($availability)
    {
        switch ($availability) {
            case '1':
                return 'В наличии';
                break;
            case '0':
                return 'Под заказ';
                break;
        }
    }

    /**
     * Возвращает путь к изображению
     * @param integer $id
     * @return string <p>Путь к изображению</p>
     */
    public static function getImage($id)
    {
        // Название изображения-пустышки
        $noImage = 'no-image.jpg';

        // Путь к папке с товарами
        $path = '/upload/images/products/';

        // Путь к изображению товара
        $pathToProductImage = $path . $id . '.jpg';
```

```
            if
(file_exists($_SERVER['DOCUMENT_ROOT'].$pathToProductImage)) {
                // Если изображение для товара существует
                // Возвращаем путь изображения товара
                return $pathToProductImage;
            }

            // Возвращаем путь изображения-пустышки
            return $path . $noImage;
        }

    }
```