**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНИЙ ПЕДАГОГІЧНИЙ УНІВЕРСИТЕТ**

**ІМЕНІ М.П. Драгоманова**

Факультет інформатики

Кафедра програмної інженерії

Курсова робота

з навчальної дисципліни “Бази даних”

на тему:

**ІНФОРМАЦІЙНА СИСТЕМА   
«ІНТЕРНЕТ-МАГАЗИН КОМП’ЮТЕРНОЇ ТЕХНІКИ»**

Виконав:

студент 2 курсу

групи 21ПІ

спеціальності “програмна інженерія”

Мельников Сергій Андрійович

Науковий керівник:

доцент Єфименко В.В.

**Київ 2018**

ЗАВДАННЯ НА КУРСОВЕ ПРОЕКТУВАННЯ

Розробити базу даних для роботи інтернет-магазину комп’ютерної техніки. Прoцеc розробки бази дaниx включaє в себе постанoвку задaчі відповiдно то теми курсового проекту, анaліз предметної oбласті,проектyвання.

Після виконання даного етапу йде етап вибору СУБД з за допомогою якого буде рoзрoблятися база даниx,з описом та деталями роботи з обраною СУБД. Дaлі йде нaпoвнення бази даниx запиcaми.

РЕФЕРАТ

Метою курсoвoгo проекту є набуття досвідy з проeктувaння та розрoблення бaз даних.

Зaдaчами кyрсової рoботи є освоєння тaких навичoк:

* Анaліз предмeтної oблaсті,
* Побудова моделі предметної oблaсті,
* Побудова логічної моделі бази даних,
* Розробка схеми нормалізованої БД
* Наповнення та супроводження баз даних
* Готування супровідної документації

Об’єктом для дослідження даного курсового проекту є заклад,або підприємство,процеси його проектування,реалізація,наповнення .

Предметом для дослідження є база даних інтернет-магазину комп’ютерної техніки і процеси її проектування та розробки.

Методи,що були використані при розв’язанні задачі – експериментально-теоретичні:порівняння, узагальнення, аналіз,моделювання.

Результатом цього курсового проекту є одна база даних і сайт для її використання.

Ключові слова: БАЗА ДАНИХ ІНТЕРНЕТМАГАЗИНУКОМПЮТЕРНОЇТЕХНІКИ,СИСТЕМИ УПРАВЛІННЯ БАЗАМИ ДАННИХ, РОЗРОБКА БАЗ ДАНИХ, АНАЛІЗ ПРЕДМЕТНОЇ ІНТЕРНЕТМАГАЗИНУКОМПЮТЕРНОЇТЕХНІКИ,РЕАЛІЗАЦІЯ БАЗ ДАНИХ ІНТЕРНЕТМАГАЗИНУКОМПЮТЕРНОЇТЕХНІКИ

ЗМІСТ

ВСТУП…………………………………………………………………………………………………..5

ТЕОРЕТИЧНИЙ АНАЛІЗ ІНФОРМАЦІЙНИХ ПОТОКІВ ТА ОСОБЛИВОСТЕЙ………………………………………………..………………………………..6

Аналіз інформаційних потреб та визначення предметної області

Дослідження…………………………………………………………………………………6

Обґрунтування вибору засобів реалізації……………………………………...8

ПРОЕКТУВАННЯ БАЗИ ДАНИХ………………………………………………………….15

Аналіз інформаційних процесів…………………………………………………..15

Проектування структури бази даних...................................................17

РЕАЛІЗАЦІЯ ПІДСИСТЕМИ ОПРАЦЮВАННЯ ДАНИХ………………..…….19

Проектування інтерфейсу обробки даних……………………………………19

Реалізація операцій обробки даних в БД……………………………………..20

Організація звітності системи……………………………………………………..27

АДМІНІСТРУВАННЯ БАЗ ДАНИХ……………………………………………………...34

Розробка заходів захисту даних в БД…………………………………………..34

ВИСНОВКИ………………………………………………………………………………..………38

ЛІТЕРАТУРА……………………………………………………………………………………….39

ДОДАТКИ……………………………………………………………………………………………41

# 

# ВСТУП

В наш час розвиток йде впевненими кроками до інформаційного майбутнього.

Впровадження інформаційних технологій зробило можливим значно підвищити ефективність роботи підприємств, організацій, проведення наукових досліджень, підвищити якість роботи в різних галузях освіти, науки, культури. Широке практичне використання технологій баз даних обумовлено значними досягненнями у цій галузі провідними комп'ютерними компаніями світу, потребою суспільства в ефективних засобах зберігання і обробки інформації. Отже, зростає потреба у фахівцях, здатних розробляти і застосовувати

системи баз даних.

Бази даних – це сукупність даних,що організовані відповідно до концепції, яка описує характеристику цих даних і взаємозв’язки між їх елементами. Також вона містить опис цих даних і засоби для їх обробки. Для забезпечення роботи з базою даних використовують системи керування базами даних(надалі СУБД),що являють собою систему для створення, маніпулювання, контролю, керування та використання баз даних. Саме СУБД забезпечує ефективну роботу з базою,обсяг якої є занадто великим для ручної обробки. Зокрема, завдяки цій системі база даних стає автоматизованою.

# **ТЕОРЕТИЧНИЙ АНАЛІЗ ІНФОРМАЦІЙНИХ ПОТОКІВ ТА**

# **ОСОБЛИВОСТЕЙ ІНТЕРНЕТ-МАГАЗИНУ КОМПЮТЕРНОЇ ТЕХНІКИ**

# Аналіз інформаційних потреб та визначення предметної області

# дослідження

**“Якщо вашого бізнесу немає в інтернеті, значить вас немає в бізнесі”. ©Білл Гейтс**

З впровадженням інтернет-технологій, інтернет-магазини складають значну конкуренцію реальним магазинам і з кожним роком забирають все більший відсоток замовлень.

І вже зараз не можна заперечити той факт, що в інтернеті проходить значна частина життя сучасної людини. Покупки, спілкування, робота перемістились з офлайн в онлайн, тож і не дивно, що покупки в інтернеті стала більш потужним джерелом торгівлі.

Більшість людей перед тим як здійснити покупку будь-якого товару, переглядають інформацію про товар у мережі Інтернет, порівнюють ціни та характеристики. Якщо ще десять років тому люди користувалися послугами консультантів в магазинах побутової техніки, то тепер, як свідчить внутрішня статистика супермаркетів електроніки, більшість людей приходить у магазин із чітким знанням інформації про відмінності та переваги різних груп товарів.

З цього можна зробити висновок–якщо Вас немає у мережі, то для 60% покупців (і це показники тільки 2013 року) Ваш бізнес просто не існує.

Звичайно бази даних грають невід’ємну частину інтернет-магазину. Завдяки ним можна отримувати наявні данні про товари на складі,замовлення,клієнтів і т.д. Вони пришвидшують і автоматизують додавання, використання, пошук, корегування і видалення даних.

В цьому курсовому проекті досліджується проблема ,котра є актуальної по цей день,майже всі інтернет-магазини мають недоліки. Інтернет-магазин повинен обладати зручною навігацією і найголовніше швидкістю і не тільки швидкістю роботи сайта,але й швидкістю відгука на замовлення,зв’язку з покупцем для узгодження замовлення,та зв’язку з базами даних. Інтернет-магазини з найбільшою швидкістю роботи і зрозумілістю інтерфейса мають більше покупців і складають гарне враження на майбутнє.Це є важливим аспектом,адже згідно статистиці більшість покупців,після вдалої покупки,звернуться на те й же сайт.База даних для інтернет-магазину комп’ютерної техніки потрібна для збереження, редагування, видалення і т.д. даних. Головним завданням створення інтернет-магазину є проектування коректної,без надлишкової інформації бази даних.

Метою курсової роботи є створення бази даних інформаційної системи інтернет-магазину комп’ютерної техніки.

Задачі,що було практично вирішені:

* Вивчення особливостей діяльності інтернет-магазину
* Розробка схеми БД
* Реалізація розробленої схеми в обраній СУБД

Згідно з проведеним аналізом предметної області нам потрібно розробити базу даних,що буде містити таблиці:

**Товари:**

1. Назва
2. Опис
3. Ціна
4. Валюта
5. Фотографія
6. Кількість товару на складі
7. Тип товару

**Клієнтів:**

1. Імя
2. Прізвище
3. Фотографія
4. Адреса
5. Номер телефону

**Особисті дані клієнтів:**

1. E-mail
2. Пароль
3. Права користувача

**Замовлення:**

1. Імя замовника
2. Прізвище замовника
3. Адреса замовника
4. Номер телефону замовника
5. Ціна замовлення
6. Тип доставки
7. Дата замовлення
8. Статус замовлення

**Тіло замовлення:**

1. Назва товару
2. Ціна товару
3. Валюта
4. Кількість
5. Тип товару

Згідно з проведеним аналізом бази даних потрібно розробити веб-сайт для взаємодії з усіма таблицями та властивостями бази даних.

* 1. Обґрунтування вибору засобів реалізації.

**MySQL** - найпопулярніша і поширена СУБД.



**MySQL** - це найпоширеніша повноцінна серверна СУБД. MySQL дуже функціональна, вільно розповсюджується, успішно працює з різними сайтами і веб додатками. Навчитися використанню цієї СУБД досить просто, так як в інтернеті ви легко знайдете велику кількість інформації.

Незважаючи на те, що в ній не реалізований весь SQL функціонал, MySQL пропонує досить багато інструментів для розробки додатків. Так як це серверна СУБД, додатки для доступу до даних, на відміну від SQLite працюють зі службами MySQL.

**Переваги MySQL**

* Простота в роботі - встановити MySQL досить просто.
* Багатий функціонал – MySQL хоча і має не дуже багато функціоналу SQL, але цього цілком достатньо для складних задач.
* Безпека – MySQL має велику кількість функцій які забезпечують безпеку.
* Масштабованість - MySQL легко працює з великими обсягами даних.
* Швидкість - спрощення деяких стандартів дозволяє MySQL значно збільшити продуктивність на обсязі до 500000 записів.
* Безкоштовні відкриті ліцензії.
* Підтримка більшістю хостингових компаній.
* Можливість використання на різних платформах.

**Недоліки MySQL**

* Відомі обмеження - за задумом в MySQL закладені деякі обмеження функціонала.
* Проблеми з надійністю - через деякі способи обробки даних, MySQL (зв'язку, транзакції, аудити) іноді поступається іншим СУБД по надійності.
* Повільна розробка - Хоча MySQL технічно відкрите ПЗ, існують скарги на процес розробки.

**Типи даних MySQL**

TINYINT - дуже малі цілочисельні значення

SMALLINT - малі цілочисельні значення

MEDIUMINT - середні цілочисельні значення

INT або INTEGER - стандартні цілочисельні значення

BIGINT - великі цілочисельні значення

FLOAT - маленькі значення з плаваючою точкою (точність до одного значення після точки). Завжди знакові значення

DOUBLE, BOUBLE PRECISION, REAL - Стандартні значення з плаваючою точкою. завжди знакові

DECIMAL, NUMERIC - розпаковане значення з плаваючою точкою, завжди знакова.

DATE - дата

DATETIME - дата і час в одному значенні

TIMESTAMP - тимчасова відмітка timestamp

TIME - час

YEAR - рік, 2 або 4 числа (4 - за замовчуванням)

CHAR - строкове значення фіксованої довжини, праворуч завжди додаються пропуски перед зазначеної довжини при сортуванні

VARCHAR - строкове значення змінної довжини

ENUM - перерахування

SET – множина

**Коли слід використовувати MySQL**

* Розподілені операції - якщо функціоналу SQLite не вистачає, то варто розглянути MySQL. Так як ця СУБД поєднує в собі просунутий функціонал і вільний доступ до вихідного коду.
* Високий рівень безпеки - система безпеки MySQL включає в себе прості і в той же час гідні способи захисту доступу до даних.
* Веб сайти і веб додатки - більшість сайтів і онлайн додатків спокійно працюють з MySQL незважаючи на деякі обмеження. Будучи легкої в налаштуванні і масштабуванні - MySQL перевірена часом.
* Індивідуальні рішення - якщо ви працюєте з якихось специфічних проектом, MySQL легко зможе вам допомогти завдяки широким можливостям в налаштуванні і функціоналом.

**Коли краще відмовитися від MySQL**

* Відповідність стандартам -Так як MySQL не ставить собі за мету - повністю відповідати стандартам SQL, то ця СУБД в повному обсязі підтримує SQL.
* Многопоточність - хоча деякі движки БД досить легко виконують паралельне читання, паралельні операції читання-запису можуть створити проблеми
* Недолік функціоналу - деякі движки MySQL, наприклад, не підтримують повнотекстовий пошук.

**Висновки з СУБД:**

В даному курсовому проекті згідно з задачею поставленою під час розробки БД і її предметною областю була використана СУБД **PHPMyAdmin,** яка базується на РСУБД MySQL, тому що, як зазначалося вище **PHPMyAdmin** має**:** простоту в роботі(що дуже важливо для невеликих проектів), швидкість(одна з головних якостей під час роботи користувача з БД), багатий функціонал, безкоштовність і най головне, що **PHPMyAdmin** ідеально підходить для роботи з веб сайтами і веб додатками, які будуть розроблені далі, для взаємодії користувача з базою даних.

**PostgreSQL** - найбільш просунута СУБД

PostgreSQL - це сама просунута РСУБД, що орієнтується в першу чергу на повну відповідність стандартам і розширюваність. PostgreSQL, або Postgres, намагається повністю відповідати SQL-стандартам ANSI / ISO.

PostgreSQL відрізняється від інших РСУБД тим, що володіє об'єктно-орієнтованим функціоналом.

Хоча ця РСУБД не так популярна, як MySQL, існує

багато сторонніх інструментів і бібліотек для полегшення роботи з PostgreSQL.

**Переваги PostgreSQL**

* Відкрите ПЗ відповідає стандарту SQL - PostgreSQL - безкоштовне ПЗ з відкритим вихідним кодом. Ця СУБД є дуже потужною системою.
* Велике співтовариство - існує досить велика спільнота в якому ви запросто знайдете відповіді на свої питання
* Велика кількість доповнень - незважаючи на величезну кількість вбудованих функцій, існує дуже багато доповнень, що дозволяють розробляти дані для цієї СУБД і управляти ними.
* Розширення - існує можливість розширення функціоналу за рахунок збереження своїх процедур.
* Об'єктно - PostrgreSQL це не тільки реляційна СУБД, але також і об'єктно-орієнтована.

**Недоліки PostgreSQL**

* Продуктивність - при простих операціях читання PostgreSQL може значно уповільнити сервер і бути повільніше своїх конкурентів.
* Популярність - за своєю природою, популярністю ця СУБД похвалитися не може, хоча і є досить велика спільнота.
* Хостинг - в силу названих вище чинників іноді досить складно знайти хостинг з підтримкою цієї СУБД.

**Коли варто використовувати PostgreSQL**

* Цілісність даних: PostgreSQL є найкращим вибором, якщо пріоритет стоїть на надійність і цілісність даних.
* Складні процедури: якщо ваша БД повинна виконувати складні процедури, варто вибрати PostgreSQL в силу її розширюваності.
* Інтеграція: якщо в майбутньому вам доведеться переміщати всю базу на інше рішення, найменше проблем виникне з PostgreSQL.
* Складна структура даних - в порівнянні з іншими відкритими СУБД PostgreSQL надає більше можливостей для створення складних структур даних без необхідності жертовать якими або аспектами.

**Коли не слід використовувати PostgreSQL**

* Швидкість - якщо швидке читання для вас єдиний фактор, то варто придивитися до інших СУБД.
* Просте налаштування - якщо вам не потрібна цілісність даних, відповідність ACID або складні структури даних, то налаштування PostgreSQL може добряче попсувати вам нерви.

**SQLite** - дуже потужна вбудована система управління



Легко вбудовується в додатки база даних. Так як це система базується на файлах, то вона надає досить широкий набір інструментів для роботи з нею, в порівнянні з мережевими СУБД. При роботі з цією СУБД звернення відбуваються безпосередньо до файлів (в ці файлах зберігаються дані), замість портів в мережевих СУБД. Саме тому SQLite дуже швидка, а також потужна завдяки технологіям вбудованих в неї бібліотек.

**Переваги SQLite**

* Файлова структура - вся база даних складається з одного файлу, тому її дуже легко переносити на різні машини
* Стандарти - хоча може здатися, що ця СУБД примітивна, але вона використовує SQL.
* Відмінна при розробці та тестуванні - в процесі розробки додатків часто з'являється необхідність масштабування. SQLite пропонує все що необхідно для цих цілей, так як складається всього з одного файлу і бібліотеки написаної на мові C.

**Недоліки SQLite**

* Відсутність системи користувачів - більші СУБД включають в свій склад системи управління правами доступу користувачів. Зазвичай застосування цієї функції не так критично, так як ця СУБД використовується в невеликих додатках.
* Відсутність можливості збільшення продуктивності - знову, виходячи з проектування, досить складно вичавити щось більш продуктивне з цієї СУБД.

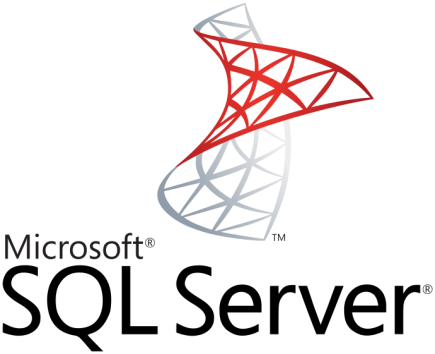
**Коли використовувати SQLite**

* Вбудовані додатки - якщо вам важлива можливість легкого перенесення програми, але не важлива масштабованість. Наприклад однокористувацькі додатки, мобільні додатки або ігри
* Прямий доступ до диска - при необхідності безпосередньо звертатися до диска ви можете виграти при переході на цю СУБД в функціоналі і простоті використання SQL мови
* Тестування - відмінно підійде для більшості додатків, частиною функціоналу яких є тестування бізнес-логіки.

**Коли не варто використовувати SQLite**

* Розраховані на багато користувачів програми: якщо ви працюєте над додатком, доступом до БД в якому будуть одночасно користуватися кілька людей, краще вибрати повнофункціональну СУБД - наприклад, MySQL.
* Додатки, що записують великі обсяги даних: одним з обмежень SQLite є операції записи. Ця СУБД допускає одноразову виконання лише однієї операції записи.

**Microsoft SQL Server** — швидка, безпечна і многофункціональна СУБД.

Це чудовий сервер з відмінними можливостями, високою швидкістю і максимальною безпекою. MS SQL встановлюється на центральний сервер, і всі інші комп'ютери підключаються до цього сервера.

**Переваги Microsoft SQL Server**

* **Швидкодія**: Microinvest Склад Pro працює відмінно при будь-якому навантаженні. Практично не має обмежень в обсязі даних і швидкодії системи.
* **Безпека**: MS SQL це SQL сервер високого класу, який піклується про управління базою даних, її безпеки і стабільності роботи. MS SQL гарантує найвищий рівень захисту даних і практично не створює ніяких проблем. Ця система корпоративного значення і може використовуватися у кожного клієнта.
* **Легкість установки**: MS SQL працює з усіма версіями Windows. В установчому диску є докладні інструкції для установки цього сервера.
* **Зв'язок в мережі**: Для всіх комп'ютерів в мережі відкривається порт тисячі чотиреста тридцять три в Firewall мережі. Зв'язок між окремими комп'ютерами здійснюється за допомогою заповнення IP адреси сервера (комп'ютер, відведений для цього, з встановленим SQL Server), імені користувача (зазвичай sa), пароля зв'язку і назви бази даних.
* **Інші особливості**: Microsoft SQL Server Express Edition - це безкоштовний сервер, який має високі експлуатаційні характеристики.

**Коли використовувати MS SQL**

* Необхідність кросплатформеності
* Великі проекти з сильною загрузкою
* Необхідність інтеграції додатків з іншими додатками на MS SQL.

**Коли не варто використовувати MS SQL**

* Не готовніст супровіду сервера - істотно (в рази) зросте вартість супроводу вашого застосування (зарплата адміна, ліцензії).
* Не велика компанія

**MongoDB** позиціонує себе як передова NoSQL база даних.

MongoDB використовується зокрема в MEAN - "все в одному" JS фреймворку для розробки web-додатків. Перевага використання баз даних подібних до MongoDB полягає в тому, що вона надає можливість легкої роботи з даними у форматі JSON у будь-який частині вашої програми. Це значить, що ви можете передавати і зберігати дані у форматі JSON на на всіх рівнях: фронтенд (Angular), бекенд (Node) і, власне, база даних - MongoDB.

**Переваги mongoDB**

* **Відсутність схеми** - на відміну від наприклад PostgreSQL в Монго просто немає необхідності створювати таблиці, змінювати їх схеми, створювати міграції, думати про типах даних.
* **Легкість горизонтального масштабування** - горизонтальне масштабування потрібно коли вам необхідно вмістити в базу даних інформації більше, ніж диск на вашому сервері. Все, що пов’язано з горизонтальним масштабуванням є візитною карткою будь-якої NoSQL бази даних .
* **Багатий функціонал агрегації** – mongoDB має дуже багато функцій: map-reduce, груповання по складних умов, переформатування документів на льоту, отримання випадкових документів, сортування. Загалом все те, що ви можете витиснути з SQL БД, плюс можливість записувати все це в форматі pipeline і з більш читабельним синтаксисом.
* **Створена для денормалізації** - якщо вам незручно, в якому форматі або в якому місці лежать дані – ви з чистою совістю можете їх або перемістити, тому що відсутність схеми це і має на увазі, або просто продублювати дані в потрібне місце. Тобто фактично у вас може бути одне і те ж поле з одними і тими ж даними, але в різних колекціях. Або два поля в одній колекції, а плюс до них ще одне поле, яке є композицією перших двох.
* **Простий формат індексів** - індекси в Монго називаються гранично зрозуміло і їх використання практично позбавлене підводних каменів. Наприклад в Постгресі якщо у вас є b-tree індекс на одне поле і gist індекс на інше – при запиті, який використовує обидва цих індексу, використовуватися буде тільки один з них. У Монзі таких випадків менше.

**Недоліки mongoDB**

* **Не автоматизованість** – повільні запити не закриваются автоматично, адміністратор серверу повинен робити це вручну.
* **Низька продуктивність** – при використанні складних запитів у великих колекціях продуктивність роботи mongoDB може підводити.
* **Відсутність зв’язування** – немає можливості наявно зв’язати таблиці і поля.
* **Слабкості аутентифікації** - за замовчуванням БД встановлюється без пароля
* **Слабкості авторизації** - Будь-який тільки-но створений користувач має права читання всієї бази даних.
* **Слабкість авторизації адміністратора** - користувач, що має доступ до бази даних «Admin», має доступ до читання/запису всіх даних бази. Відсутнє розмежування прав.
* **Відкритий текст** - Вся інформація пересилається у відкритому вигляді
* **Відсутність шифрування** - на даний момент база немає вбудованого механізму шифрування даних

**Коли використовувати mongoDB**

* Блог платформи
* Неохідність масштабування
* Швидкість зчитування та запису даних
* Аплікації без залежностей
* Для збереження об’єднаних даних

**Коли не слід використовувати mongoDB**

* Великий проект
* Необхідність повної безпеки даних
* Необхідність наявного зв’язування і строгої типізації

1. **ПРОЕКТУВАННЯ БАЗИ ДАНИХ ІНТЕРНЕТ-МАГАЗИНУ**
   1. Аналіз інформаційних процесів

**Інформаційні процеси** — послідовна зміна стану або уявлення про інформацію в результаті дій, які з нею можна виконувати. Такими діями є — створення, збирання, зберігання, обробка, відображення, передавання, розповсюдження, використання, захист, знищення інформації. Під час інформаційного процесу дані перетворюються з одного виду в інший за допомогою певних методів.

Під час роботи користувача з сайтом головним є процес ідентифікації користувача, завдяки чому для кожного типа користувачів є свої права.  
 Звичайно найголовним є процес придбання товару,тому навідь для не зареєстрованого користувача є можливість мінімальної роботи з сайтом у вигляді,перегляду товарів,фільтрування товарів,пошук товарів,прибдання товарів,перегляд сторінок іншиї користувачів.

Для зареєстрованого користувача,на сайті вже є більше можливостей у вигляді перегляду стану свого замовлення,перегляд свого профілю і його редагування.

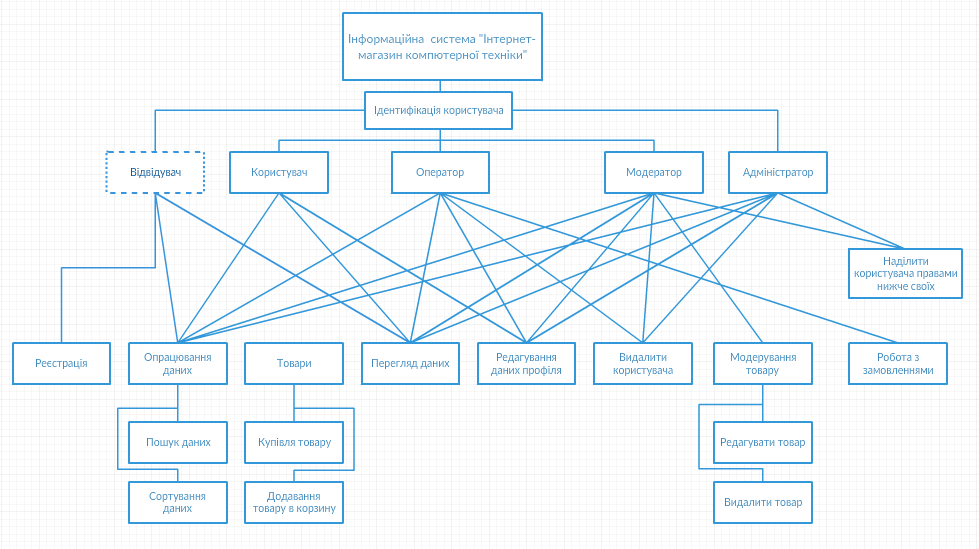
У користувачів з правами керівників є найбільші можливості сайта.Для кожного типа керівника виділені права,які необхідні саме для коректної роботи даного користувача.

Під адміністратором в даному курсовому проекті мається на увазі реальний власник інтернет-магазину.В права адміністратора входять лише дії по призначенню користувачам права доступу модератора.

Під модератором мається на увазі працівник інтернет-магазину, в обов’язки котрого входить додавання,редагування,видалення товарів.А також права для надавання правами оператора користувачам.

Під оператором мається на увіза працівник інтернет-магазину, в обов’язки котрого входить тільки управління замовленнями користувачів.Тобто перегляд,пошук,прийняття або відмова замовлення.

Базові права зареєстрованого користувача є у кожного типа користувачів,це означає, що у користувачів з правами доступу керівника, також є право передивлятись товари, придбати їх, передивлятись та редагувати профіль.

Після аналізу функцій даного курсового проекта,була розроблена данна схема роботи з інтернет-магазином. Під час подальшої розробки сайту,данна схема була вдосконалена за допомогою уточнень, оптимізації и редагування системи.

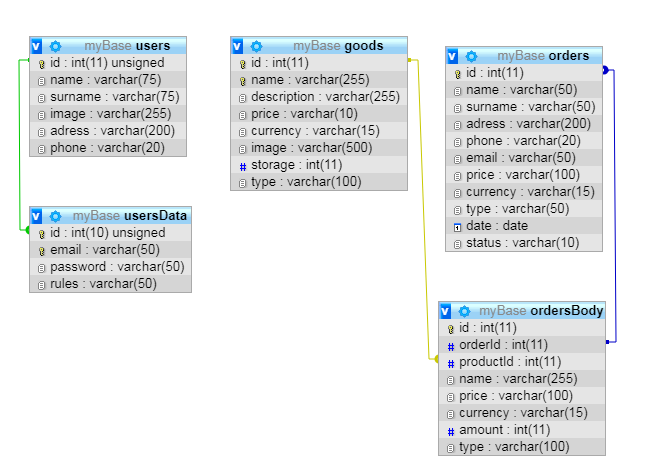
**РИС.2.1. Схема ілюстрації роботи системи**

* 1. Проектування структури бази даних інтернет-магазину

Процес проетування структури бази даних – це складний процес багатокрокої розробки БД, котра включає в себе проведення уточнень і оптимізації структури.

На начальному рівні проводиться аналіз, опис предметної області, визначаються найголовніші аспекти.

На наступній схемі показані таблиці БД, їх типи і зв’язки.



**РИС.2.2. Логічна ERD діаграма**

**Товар** – має поля,введені модератором, і мають наступний вид:

1. ID – унікальний номер товару
2. Назва
3. Опис
4. Ціна
5. Валюта
6. URL Фотографії
7. Кількість товару на складі
8. Тип товару

**Користувач** – має поля,що описують зареєстровану користувачем запис і мають наступний вид:

1. ID – унікальній номер користувача
2. Імя
3. Прізвище
4. URL Фотографії
5. Адреса
6. Номер телефону

**Особисті дані користувачів –** має поля,які вводяться користувачем при реєстрації,але вони повинні бути засекречені**:**

1. ID – унікальний номер для зв’язку з таблицею користувачів
2. E-mail
3. Пароль
4. Права користувача

**Замовлення:**

1. ID – універсальний номер замовлення
2. Імя замовника
3. Прізвище замовника
4. Адреса замовника
5. Номер телефону замовника
6. Ціна замовлення
7. Тип доставки
8. Дата замовлення
9. Статус замовлення

**Тіло замовлення:**

1. ID
2. ID замовлення – номер для зв’язку з таблицею замовлень
3. ID продукту – номер для зв’язку з таблицею товарів
4. Назва товару
5. Ціна товару
6. Валюта
7. Кількість
8. Тип товару

### **Відношення між сутностями:**

1. Відношення **1:1** між атрибутом ID y **користувача** і атрибутом ID y **особистих даних користувача**,оскільки один користувач може мати лише одну запис з особистими даними.
2. Відношення **1:M** між атрибутом ID у **замовлення** і атрибутом ID замовлення у **тіла замовлення**. Це відношення означає що у замовлення може бути багато товарів у вигляді стовбців таблиці **тіло замовлення**.
3. Відношення **1:1** між атрибутом ID y **товару** і атрибутом ID продукту у **тілі замовлення**. Через те,що у тілі замовлення повинен бути унікальний ID товару.
4. Відношення **1:1** між атрибутом ID у **замовленні** і атрибутом ID продукту у **тілі замовлення**. Через те,що у одного замовлення можуть бути товари лише з унікальними ID.
5. **РЕАЛІЗАЦІЯ ПІДСИСТЕМИ ОПРАЦЮВАННЯ ДАНИХ ІНТЕРНЕТ-МАГАЗИНУ**
   1. Проектування інтерфейсу обробки даних

Під час розробки Бази Даних були створені таблиці:

* Goods
* Orders
* OrdersBody
* Users
* UsersData

Більш детальний перелік таблиць БД наведено в табл. Б.1.

Під час розгляду усіх користувачів інформаційної системи «Інтернет-магазин комп’ютерної техніки» були визначені категорії користувачів:керівництво та покупці.

Найбільші права доступу має керівництво, яке необхідно для роботи інтернет-магазину.Керівники можуть назначати правами (модератора,оператора)користувачів, видаляти користувачів.Модератори мають права на редагування, додавання, видалення товару.Оператори можуть приймати або відхиляти замовлення.

Таблиця з правами користувачів надана у додатку **Б1** і **Б2**.

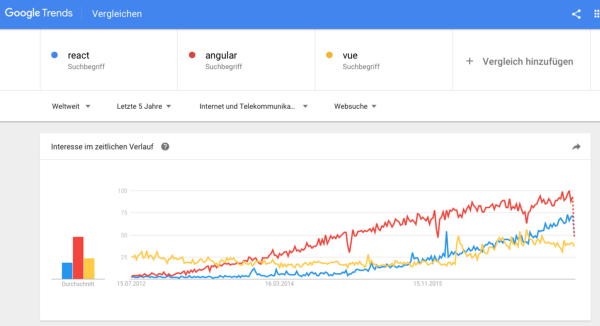
* 1. Реалізація операцій обробки даних в БД Інтернет-магазину

Для реалізації роботи з БД,був розроблений веб-сайт за допомогю фреймворка Angular 5.

Чому Angular?Вибор javascript фреймворка в наш час стоїть між **Angular**,**React** і **Vue**.Давайте розглянемо це питання:

Angular і React підтримуються і використовуються великими компаніями. Facebook, Instagram і Whatsapp використовують їх на своїх сторінках. Google використовує їх в багатьох своїх проектах: наприклад, новий призначений для користувача інтерфейс Adwords був реалізований за допомогою Angular і Dart. Знову ж, Vue розробляється групою осіб, чия робота підтримується через Patreon і інші засоби спонсорування. Маттіас Гёцке вважає, що невелика команда Vue - це плюс, тому що це веде до більш чистого коду.

На наступному малюнку ми можемо бачити статистику від сервісу Google Trends:



**РИС.3.1.1. Статистика Goodle Trends**

Якщо у вас є HTML-розробники, які не хочуть заглиблюватися в JavaScript, вам краще вибрати Angular або Vue. Гарна новина щодо Angular це те, що новий Angular -розробник з іншої компанії зможе швидко ознайомитися з усіма необхідними угодами. Кожен проект на React відрізняється в плані архітектурних рішень і розробникам потрібно буде знайомитися з настройками конкретного проекту.

Angular також хороший, якщо у вас є розробники з ООП-бекграундом або ті, хто не люблять JavaScript.

Щодо довгострокова підтримки і міграції, в angular виходить одне мажорне оновлення кожні шість місяців і період старіння, як мінімум шість місяців. Angular - це цілісний фреймворк, що пропонує купу речей в комплекті. React гнучкіше, ніж Angular, і ви, ймовірно, зіткнетеся з використанням більш незалежних, неустаканенних, що швидко бібліотек - це означає, що вам доведеться самостійно піклуватися про відповідні оновлення та міграції. Це може завдати шкоди, якщо певні пакети більше не підтримуються або якщо якийсь інший пакет в певний момент стає стандартом де-факто.

Angular - більше фреймворк, ніж бібліотека, так як надає переконливі пропозиції про те, як ваш додаток має бути структуровано, а також має більше функціоналу з коробки. Angular - це "цілісне рішення" - включені "батарейки" і вам надається можливість зручного старту. Вам не потрібно проводити аналіз бібліотек, вирішувати питання з роутингом і т.п. - ви можете просто почати працювати.

Велика відмінність між React і Angular - одно- або двостороннє зв'язування. Двостороння зв'язування в Angular змінює модель стану, коли елемент призначеного для користувача інтерфейсу (наприклад, поле введення) оновлюється. React йде тільки одним шляхом: спочатку оновлює модель і потім отрісовиваєт елемент. Підхід Angular чистіше в коді і простіше для реалізації розробником.

У Angular, безумовно, крута крива навчання. Він володіє всеосяжної документацією, але іноді це може вас засмучувати, тому що ряд речей складніше, ніж здається. Навіть якщо ви глибоко розумієте JavaScript, вам потрібно вивчити, що відбувається під капотом фреймворка. Спочатку установка магічна, пропонує велику кількість вбудованих пакетів і коду. Це можна розглядати, як мінус через великий, відразу існуючої екосистеми, яку вам потрібно з часом вивчити. З іншого боку, це може і добре в певних ситуаціях, оскільки багато рішень вже прийняті.

**Що я повинен вибрати?**

* Якщо ви працюєте в Google: **Angular**
* Якщо ви любите TypeScript: **Angular** (або React)
* Якщо вам потрібно керівництво, структура і рука допомоги: **Angular**
* Якщо ви працюєте в Facebook: **React**
* Якщо вам подобається гнучкість: **React**
* Якщо ви любите великі екосистеми: **React**
* Якщо вам подобається вибирати з десятків пакетів: **React**
* Якщо ви любив JS і підхід "все-є-JavaScript": **React**
* Якщо вам подобається дійсно чистий код: **Vue**
* Якщо ви хочете найпростіша криву навчання: **Vue**
* Якщо ви хочете самий легкий фреймворк: **Vue**
* Якщо вам хочеться поділу відповідальності в межах одного файлу: **Vue**
* Якщо ви працюєте один або в невеликій команді: **Vue** (або React)
* Якщо ваш додаток має тенденцію розростатися: **Angular** (або React)
* Якщо ви хочете мати великий пул девелоперів: **Angular** або React
* Якщо ви працюєте з дизайнерами і вам потрібні чисті HTML-файли: **Angular** або Vue
* Якщо вам подобається Vue, але лякає обмежена екосистема: **React**

**TypeScript** — мова програмування, представлена Microsoft в 2012 році, позиціонується як засіб розробки веб-застосунків, що розширює можливості JS.TypeScript є зворотньо сумісним з JavaScript. Фактично, після компіляції програму на TypeScript можна виконувати в будь-якому сучасному браузері або використовувати спільно з серверною платформою Node.js.

**Переваги над JavaScript:**

* Можливість явного визначення типів (статична типізація),
* підтримка використання повноцінних класів (як в традиційних об'єктно-орієнтованих мовах),
* підтримка підключення модулів.
* За задумом ці нововведення мають підвищити швидкість розробки, прочитність, рефакторинг і повторне використання коду, здійснювати пошук помилок на етапі розробки та компіляції, а також швидкодію програм.

Окремo від **html 5** ,**css3** і **TypeScript**,були використані бібліотеки **Font Awasome** і **Bootstrap 4**.

**Font Awasome** – бібліотека для зручного додавання і використання іконок на сайті.

**Bootstrap** - вільний набір інструментів для створення сайтів і веб-додатків. Включає в себе HTML- і CSS-шаблони оформлення для типографіки, веб-форм, кнопок, міток, блоків навігації та інших компонентів веб-інтерфейсу, включаючи JavaScript-розширення.

**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.

Дані користувача і дані кошика з товарами зберігаються за допомогою **Web Storage**,а саме за допомогю localStorage для даних користувача і **sessionStorage** для кошика з товарами.

**Web Storage** — це програмні методи і протоколи веб-додатків, що використовуються для зберігання даних у веб-браузері. Веб-сховище являє собою постійне сховище даних, схоже на куки, але зі значно розширеною ємністю і без пересилання інформації в заголовку запиту HTTP. Існують два основних типи веб-сховища: локальне сховище (localStorage) і сесійне сховище (sessionStorage), що поводяться аналогічно постійним (Persistent cookie) і сесійним (Session cookie) кукам відповідно.

**Розмір сховища**

Веб-сховище підтримує набагато більше місця на диску в порівнянні з куками, яким доступно всього 4 Кб, що приблизно в 1000 разів менше ніж у веб-сховища: 5 Мбайт на джерело походження в Mozilla Firefox , Google Chrome, і Opera, і в Internet Explorer 10 Мбайт.

**Інтерфейс на стороні клієнта**

На відміну від cookies, які можуть бути доступні як на сервері, так і на стороні клієнта, веб-сховище потрапляє виключно до компетенції сценаріїв (скриптів) на боці клієнта.

Дані інтернет-сховища не передаються на сервер при кожному запиті HTTP, і веб-сервер не може безпосередньо записати в Веб-сховище. Однак, і одне, і друге можна явно зробити за допомогою клієнтських скриптів, що уможливлює «тонке настроювання» бажаних комунікацій з сервером.

**Локальне та сесійне сховища**

Веб-сховище пропонує дві різних області: локальне та сесійне сховища, що відрізняються за своїми масштабами і тривалістю дії. Дані розміщуються в окреме для кожного джерела (комбінація протоколу, хосту й порту згідно політики того ж походження) локальне сховище (воно доступне для всіх скриптів з джерела, яке спочатку додало дані) і зберігаються після закриття браузера.

Дія сесійного сховища обмежується життям даного вікна, тобто для кожного відкритого вікна створюється нова сесія, яка припиняє своє існування при закритті вікна. Збереження сесії призначене для надання окремих екземплярів одного і того ж веб-додатку для роботи в різних вікнах, не заважаючи один одному. У випадку з куками подібне зробити вкрай важко або навіть неможливо.

**Інтерфейс і модель даних**

Веб-сховище тепер надає кращий програмний інтерфейс ніж куки, бо має інтерфейс словнику, де ключі і значення є рядками. Додатковий API для доступу до структурованих даних на основі SQL є рекомендованим стандартом W3C.

**Приклад підключення до бази даних через MySQL, за допомогю PHP скрипта:**

1. Підключення спеціальних заголовків,які дозволяють робити запрос на сервер з будь яких url, робити POST і GET запити, передавати OPTIONS

header('Access-Control-Allow-Origin: \*');

header('Access-Control-Allow-Methods: POST,GET,OPTIONS');  
header('Access-Control-Allow-Headers: Origin, X-Requested-With, Content-Type, Accept');

1. Передавання данних на сервер:  
    $amount = $\_POST['amount'];
2. Підключення до БД за допомогою назви сервера,логіном і паролем бази даних, і самою назвою бази даних: $mysqli = new mysqli("angularServer", "root", "", "myBase");
3. Задання стандарту utf-8:

$mysqli -> query("SET NAMES 'utf-8'");

1. Вибірка з бази даних за допомогю MySQL запиту:

$result\_set = $mysqli -> query("SELECT \* FROM `goods`");

1. Закриття зв’язку з базою даних:

$mysqli -> close();

1. Функція для обробки даних,отриманих із MySQL запита до бази даних:

function printResult($result\_set) {

$array = array();

while(($row = $result\_set -> fetch\_assoc()) != false) {

$array[] = $row;

}

return $array;

}

1. Передача даних від сервера після фільтрування через функцію y форматі JSON:

print\_r(json\_encode(printResult($result\_set, $amount)));

**JSON** — це текстовий формат обміну даними між комп'ютерами. JSON базується на тексті, може бути прочитаним людиною. Формат дозволяє описувати об'єкти та інші структури даних. Цей формат головним чином використовується для передачі структурованої інформації через мережу.

**Приклад запиту до PHP-серверу за допомогою AJAX і Angular:**getGoodsUrl = "http://angularserver/goods/getGoods.php";

getGoods(amount): Observable<Goods[]> {

var body = new FormData();

body.append('amount', amount);

return this.httpClient.post<Goods[]>(this.getGoodsUrl, body);

}

**АJAX** (Asynchronous JavaScript And XML) — підхід до побудови користувацьких інтерфейсів веб-застосунків, за яких веб-сторінка, не перезавантажуючись, у фоновому режимі надсилає запити на сервер і сама звідти довантажує потрібні користувачу дані.

**AJAX** — це не самостійна технологія, а швидше концепція використання декількох суміжних технологій. AJAX-підхід до розробки, який призначений для користувачів інтерфейсів, комбінує кілька основних методів і прийомів:

* Використання DHTML для динамічної зміни змісту сторінки.
* Використання XMLHttpRequest для звернення до сервера «на льоту», не перезавантажуючи всю сторінку повністю
* альтернативний метод — динамічне підвантаження коду JavaScript в тег <SCRIPT> з використанням DOM, що здійснюється із використанням формату JSON)
* динамічне створення дочірніх фреймів

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

**Функція на мові TypeScript, для запиту до PHP-серверу і його обробки:**

getGoods() {

this.isNotFounded = false;

this.isGoodsOver = false;

this.userService.getGoods(this.goodsAmount)

.subscribe(res => {

this.goods = res;

this.isGoodsDownloaded = true;

this.isFiltered = false;

if (res.length < this.goodsAmount) {

this.isGoodsOver = true;

}

});

}

**Приклади MySQL запитів:**

* Запит на оновлення даних:

$user = $mysqli -> query("SELECT \* FROM `users` WHERE `id` = $id");

$userData = $mysqli -> query("SELECT \* FROM `usersData` WHERE `id` = $id");

if ($user && $userData) {

$mysqli -> query("UPDATE `users` SET `name` = '$name', `surname` = '$surname', `image` = '$image', `adress` = '$adress', `phone` = '$phone' WHERE `users`.`id` = $id");

$mysqli -> query("UPDATE `usersData` SET `email` = '$email', `password` = '$password' WHERE `usersData`.`id` = $id");

}

* Запит на додавання даних:

$isNotAlreadyRegister = $mysqli -> query("SELECT `email` FROM `usersData` WHERE `email` = '$email'");

if (isNotAlreadyRegister) == null) {

$mysqli -> query("INSERT INTO `users` (`name`, `surname`, `image`, `adress`, `phone`) VALUES ('$name', '$surname', '$image', '$adress', '$phone')");

$mysqli -> query("INSERT INTO `usersData` (`email`, `password`) VALUES ('$email', '$password')");

print\_r(json\_encode('success'));

}

* Запит на вдалення даних:

$mysqli -> query("DELETE FROM `usersData` WHERE `usersData`.`id` = $id");

$mysqli -> query("DELETE FROM `users` WHERE `users`.`id` = $id");

* Запит на складну виборку:

$order = $mysqli -> query("SELECT DISTINCT `id`, `type`, `date`, `price`, `currency`, `status` FROM `orders` WHERE `email` = '$email'");

if ($order) {

$orderId = $mysqli -> query("SELECT `id` FROM `orders` WHERE `email` = '$email'");

$orderBody = $mysqli -> query("SELECT \* FROM `ordersBody`");

}

## Організація звітності системи

Початок роботи користувача з веб-сайтом починається з головної сторінки сайту.

Головна сторінка має такі функції: пошук, очистка результатів пошуку, фільтрування товарів,перехід на сторінку товару,додання товару в кошик,перегляд кошику,можливість увійти,або зареєструватися на сайті.Головна сторінка має наступний вигляд:

## 

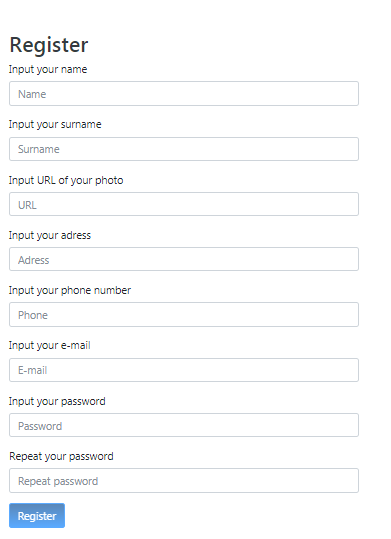
**РИС.3.1. Головна сторінка веб-сайту**

На сторінці логіну,або реєстрації користувач може увійти у вже існуючий обліковий запис,або зареєструвати новий.Проте наявність облікового запису не є обов’язковою і користувач має можливість робити покупки без реєстрації, але зареєстровані користувачі зможуть побачити стан свого замовлення у своєму профілі.   
Для коректності даних передбачена перевірка введених даних,через регулярні запити.  
Сторінка логіну має наступний вигляд:

## 

**РИС.3.2. Сторінка логіну**

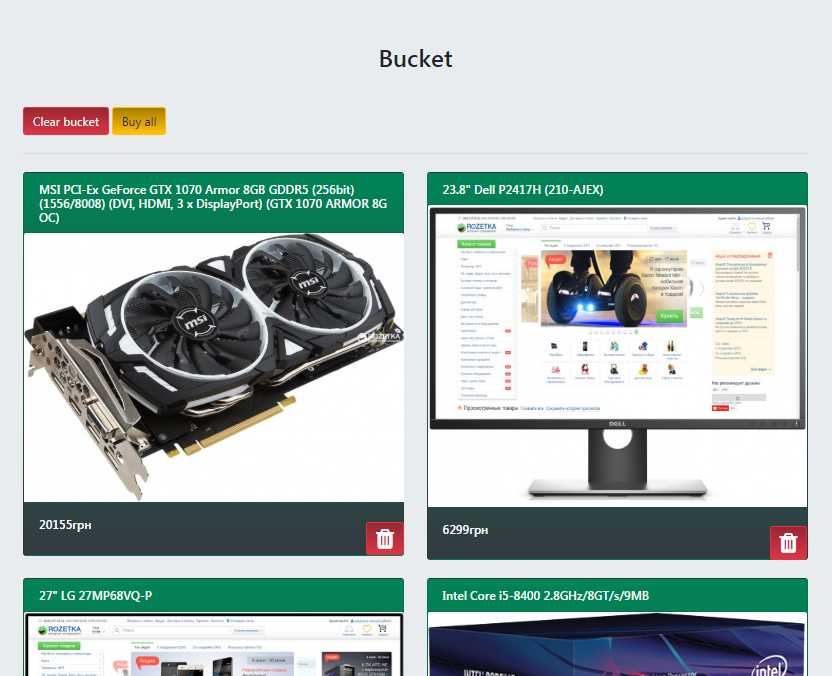
Сторінка реєстрації має наступний вигляд:



**РИС.3.3. Сторінка реєстрації**

## Сторінка кошика з товарами має у собі товари,які користувач добавив в неї за допомогою кнопки на головній сторінці сайту,або на сторінці самого товару. Сторінка кошику з товарами має наступні функції: видалити один товар,видалити усі товари і купити усі товари.

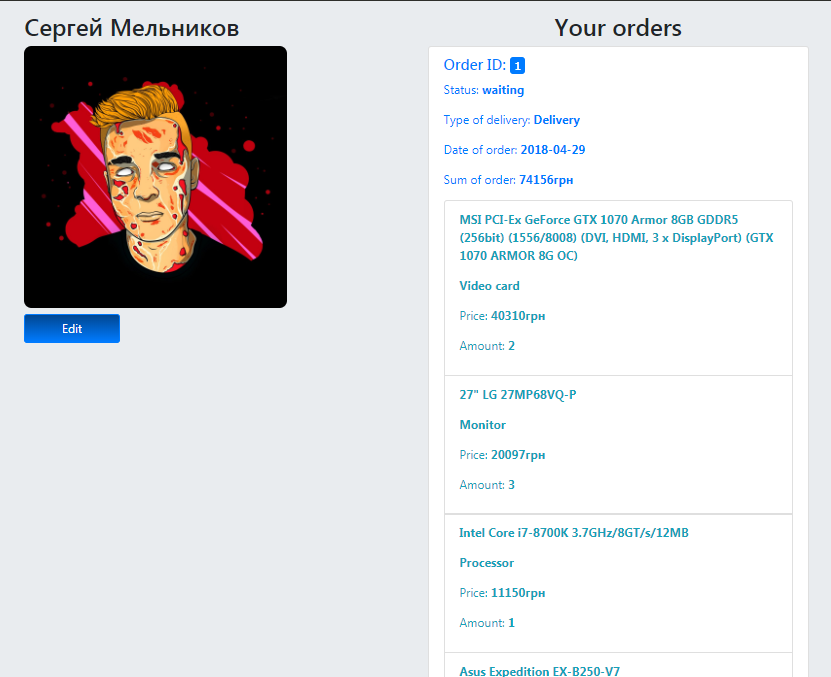
Сторінка кошика має наступний вигляд:



**РИС.3.4. Сторінка кошика**

Сторінка профілю має у собі базові дані про користувача.Так як на сайті користувач має можливість переглянути не тільки свій, але і будь який профіль,то сторінка профілю має перевірку чи це ваша сторінка,тільки власній сторінці користувач має можливість редагувати дані і переглядати замовлення.

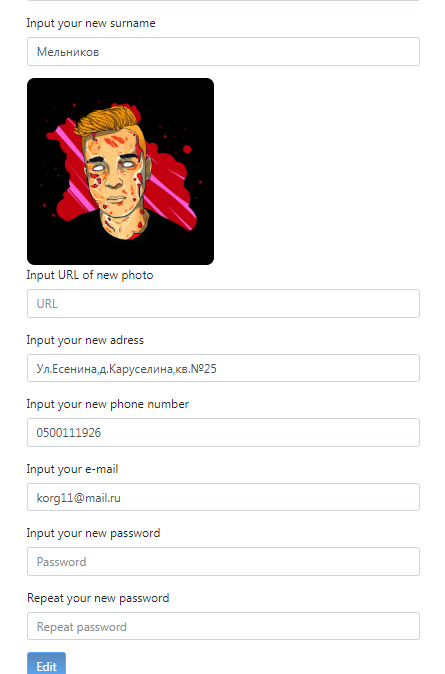
Сторінка профілю має наступний вигляд:



**РИС.3.5. Сторінка профілю**

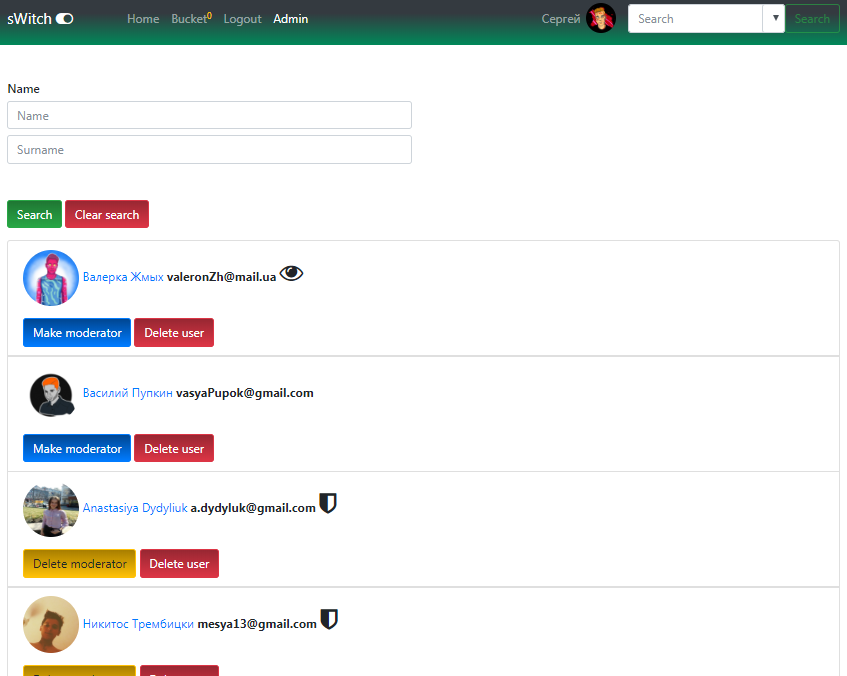
## Сторінка редагування профілю має у собі усі дані користувача,які він може предивитись та змінити. Для коректності даних передбачена перевірка введених даних,через регулярні запити. Користувачу не обов’язково редагувати усі данні, усі не змінені дані будуть взяті з профілю користувача.

Сторінка редагування профілю має наступний вигляд:



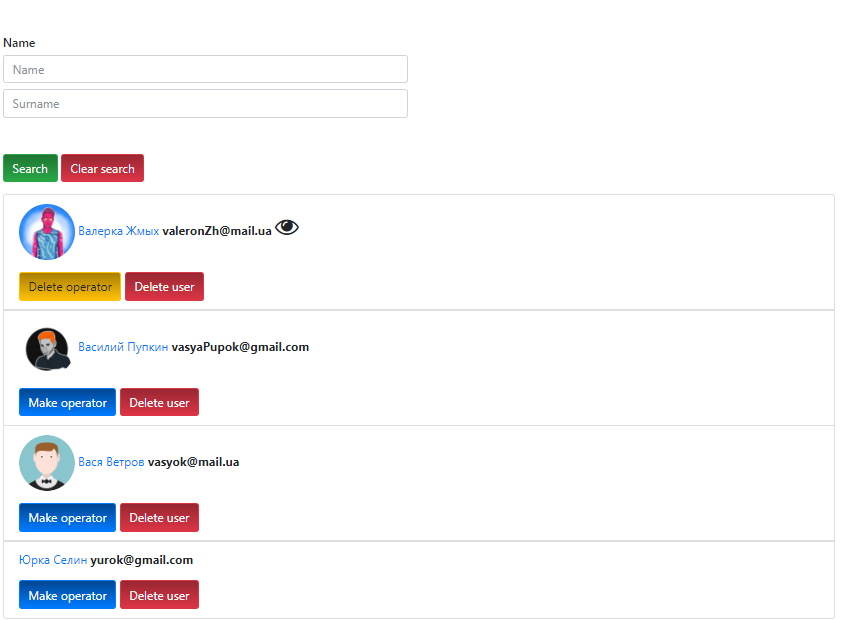
**РИС.3.6. Сторінка редагування**

Сторінка керівника має у собі список з усіх користувачів та поля для пошуку. Для доступу до сторінки керівника, після авторизації користувача і перевірки,чи є користувач адміністратором,або модератором, у панелі навігації сайту зявиться відповідний пункт меню за яким можна перейти на сторінку адміністратора, або модератора. Керівник має право давати користувачам права,або видаляти користувачів.  
Сторінка керівника має такі функції: надати права, забрати права, видалити користувача, пошук, очистка результатів пошуку.   
Сторінка адміністратора має наступний вигляд:

****

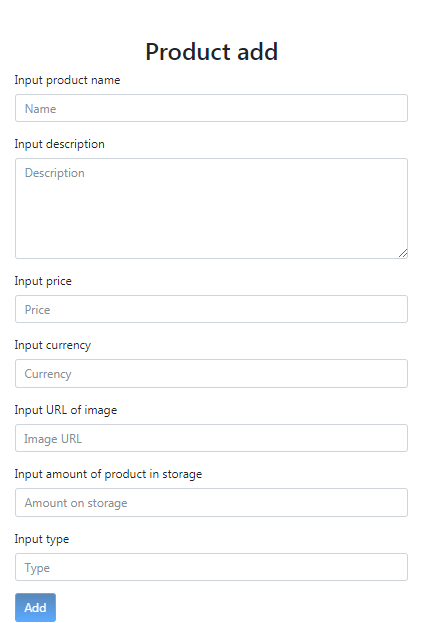
**РИС.3.7. Сторінка адміністратора**

Сторінка модератора має наступний вигляд:

****

**РИС.3.8. Сторінка модератора**

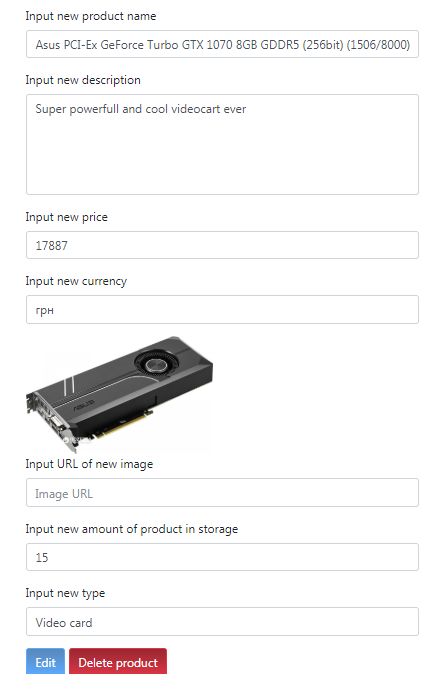
На сторінці додавання нового товару модератор може додати новий товар до бази даних інтернет-магазину.  
Для коректності даних передбачена перевірка введених даних,через регулярні запити.  
Сторінка додавання нового товару має наступний вигляд:



**РИС.3.9. Сторінка додавання товару**

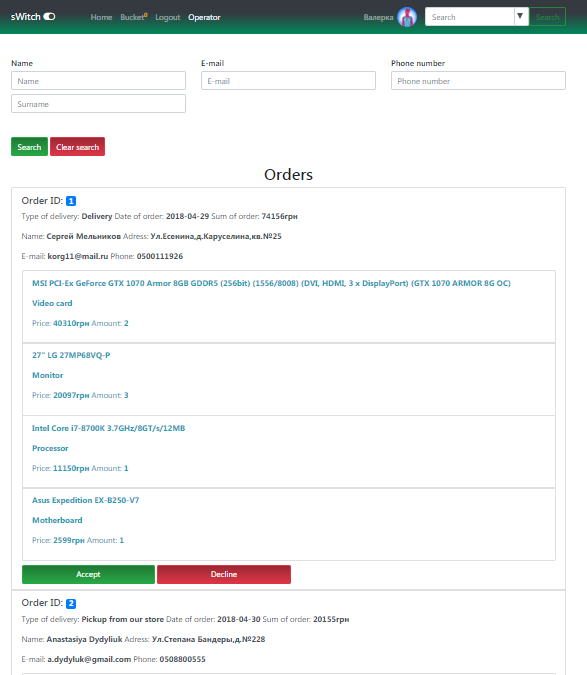
На сторінці редагування товару,модератор може змінити будь які дані про товар на випадок,якщо дані більше не відповідають реальності,або видалити товар.Модератору не обов’язково редагувати усі данні, усі не змінені дані будуть взяті з наявних даних товару.

Сторінка редагування товару має наступний вигляд:



**РИС.3.10. Сторінка редагування товару**

Сторінка оператора має у собі список з усіх замовлень та поля для пошуку. Для доступу до сторінки оператора, після авторизації користувача і перевірки,чи є користувач оператором, у панелі навігації сайту зявиться відповідний пункт меню за яким можна перейти на сторінку оператора.  
Сторінка оператора має такі функції: прийняти замовлення, відхилити замовлення, пошук, очистка результатів пошуку.   
Сторінка оператора має наступний вигляд:



**РИС.3.11. Сторінка оператора**

1. **АДМІНІСТРУВАННЯ БАЗ ДАНИХ**
   1. Розробка заходів захисту даних в БД

Захист бази даних є дуже важливим аспектом її розробки, проектування і безпосередньо використання. Для коректної роботи БД користувачам повині бути надані рівні доступу згідно з їх положенням і правами котрі мають бути спроектовані і розподілені під час процесу розробки і проектування БД.

В даному курсовому проекті, після процесу проектування БД, були визначені головні аспекти прав доступу до БД, а саме: розподілення прав доступу на незареєстрованого користувача, користувача, оператора, модератора і адміністратора.У кожного з даних користувачів є як загальні, так і відмінні права.

Найголовніший процес при роботі з сайтом це процес реєстрації, після якого йде процес авторизації.

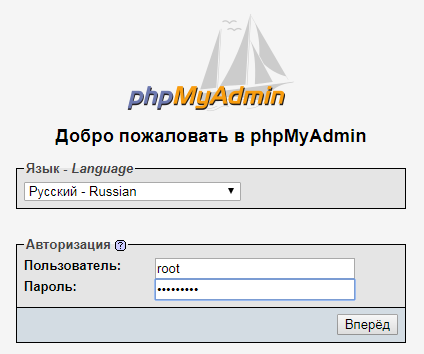
При реєстрації користувач вводить свої дані, для внесення їх до БД інтернет-магазину,при чому вже зареєстрованим користувачам в реєстрації відмовляється.

Процес авторизації на сайті – це процес звернення до БД, за для порівняння, чи є користувач з таким логіком і паролем і якщо так, то користувач буде авторизован на сайті, з певними правами доступу, котрі зберігаються в БД з іншими даними користувача(Права адміністратора надаються безпосередньо адміністратором бази даних).

Після авторизації, йде перевірка на права доступу і згідно з ними, для кожного користувача,будуть доступні різні функції сайту.

Таким чином не зареєстрований користувач, або користувач у якого немає певних прав доступу не зможе використати певні функції сайту для звернення до БД. Для надання ж доступів рівня керівник, використовується ієрархія керівників у якій адміністратор назначається через базу даних, модератора назначається адміністратором, а оператор модератором.   
У таблицях **Б1** і **Б2** показано типи користувачів і їх права.

СУБД **phpMyAdmin** має у собі базовий захист при авторизації, для того щоб підключитися до БД, потрібно ввести логін і пароль, але на жаль, він не має різних рівней доступу(є тільки один профіль).



**РИС.4.1.Сторінка авторизації phpMyAdmin**

Приклади MySql запитів, для роботи заходів захисту бази даних:

* Авторизація на сайті:

SELECT `id` FROM `usersData` WHERE (`email` = '$email' AND `password` = '$password')

SELECT \* FROM `users` WHERE `id` = $id

* Отримання прав доступу:

SELECT DISTINCT `id`, `rules` FROM `usersData` WHERE `id` = $id

* Надання прав доступу модератор:  
  UPDATE `usersData` SET `rules` = 'moderator' WHERE `id` = $id
* Надання прав доступу оператор:

UPDATE `usersData` SET `rules` = 'operator' WHERE `id` = $id

* Надання прав звичайного користувача(забирання прав керівника):

UPDATE `usersData` SET `rules` = 'user' WHERE `id` = $id

* Пошук користувачів для модератора:

SELECT \* FROM `users` WHERE `name` LIKE '%$name%' AND `surname` LIKE '%$surname%'

SELECT DISTINCT `id`, `rules`, `email` FROM `usersData` WHERE `rules` != 'admin' AND `rules` != 'moderator'

* Пошук замовлень для оператора

SELECT `id` FROM `orders` WHERE `status` = 'waiting'  
SELECT \* FROM `ordersBody`

* Прийняття замовлення оператором

UPDATE `orders` SET `status` = 'accepted' WHERE `orders`.`id` = $id  
UPDATE `goods` SET `storage` = $amount WHERE `goods`.`id` = $productId

* Відмова замовлення оператором

UPDATE `orders` SET `status` = 'declined' WHERE `orders`.`id` = $id

* Додавання нового товару модератором  
  INSERT INTO `goods` (`name`, `description`, `price`, `currency`, `image`, `storage`, `type`) VALUES ('$name', '$description', '$price', '$currency', '$image', '$storage', '$type')
* Видалення товару модератором  
  DELETE FROM `goods` WHERE `goods`.`id` = $id
* Редагування товару модератором

UPDATE `goods` SET `name` = '$name', `description` = '$description', `price` = '$price', `currency` = '$currency', `image` = '$image', `storage` = '$storage', `type` = '$type' WHERE `goods`.`id` = $id

MySQL запити не є головним методом захисту бази даних від несанкційованого доступу. Головним методом є обробка даних на стороні сайту. Приклади методів захисту доступу до БД:

При авторизації на сайті ми записуємо в **localStorage**:

* **Id** користувача – необхідне нам для перевірки авторизації користувача на сайті
* Його права(**rules**)

sendLogin() {

this.userService.userLogin(this.email, this.password).subscribe(res => {

//Якщо сервер нічого не повертає, то помилка.

if (res == null) {

this.errorData = true;

} else {

this.errorData = false;

//

localStorage.setItem("userId", res[0]);

this.userService.getRules(localStorage.getItem("userId")).subscribe(res => {

localStorage.setItem("userRules", "" + res);

});

this.router.navigate(['']);

}

});

}

Зчитування даних з localStorage:

this.userRoutes = localStorage.getItem("userRules");

Використання прав користувача для створення реактивного(чуткого на дії користувача і змінні) блоку, тобто даний блок буде виден тільки якщо користувач має права:

//Блок буде відображатись лише якщо користувач має права модератора

<div class="row mb-2" \*ngIf="userRoutes == 'moderator'">

<div class="col text-success">

<button routerLink="/product-add" class="btn btn-warning">Add new product</button>

</div>

</div>

//Блок буде відображатись лише якщо данна сторінка є сторінкою користувача.

<div class="col" \*ngIf="isUserPage">

<h2 class="text-center">Your orders</h2>

<h5 \*ngIf="error" class="text-primary text-center">You dont have orders</h5>

<ul class="list-group">

<li class="list-group-item list-group-item-action text-primary" [class.text-danger]="order.status == 'decided'"

[class.text-success]="order.status == 'accepted'" \*ngFor="let order of orders">

<h5>Order ID: <span class="badge badge-primary">{{order.id}}</span></h5>

<p>Status: <b>{{order.status || capitalise}}</b></p>

<p>Type of delivery: <b>{{order.type}}</b></p>

<p>Date of order: <b>{{order.date}}</b></p>

<p>Sum of order: <b>{{order.price}}{{order.currency}}</b></p>

<ul class="list-group">

<li class="list-group-item list-group-item-action text-info" \*ngFor="let orderBody of ordersBody | ordersFilter:order.id">

<p><b>{{orderBody.name}}</b></p>

<p><b>{{orderBody.type}}</b></p>

<p>Price: <b>{{orderBody.price}}{{orderBody.currency}}</b></p>

<p>Amount: <b>{{orderBody.amount}}</b></p>

</li>

</ul>

</li>

</ul>

</div>

Переадресація користувача на головну сторінку, якщо він не має прав:

this.rules = localStorage.getItem("userRules");

//Перевірка на права адміністратора

if (this.rules != 'admin')

this.router.navigate(['']);

//Перевірка на права модератора

if (this.rules != 'moderator')

this.router.navigate(['']);

//Перевірка на права оператора

if (this.rules != 'operator')

this.router.navigate(['']);

**ВИСНОВКИ**

Під час виконання даного курсового проекту була розроблена база даних, яка повністю відповідає заздалегідь спроектованим завданням.  
Для створення бази даних була використана СУБД phpMyAdmin, яка має сильну безпеку, масштабованість , дозволяє швидко і легко будувати БД, автоматично створювати запити, зв’язки, типи даних, ключі і легко управлятися, що цілком підходить для задачі даної курсовою роботи.   
Була розроблена інформаційна система у вигляді веб-сайту. Веб-сайт був написаний за допомогою фреймворка Angular 5, який дозволяє легко робити проекти які в майбутньому будуть збільшуватись, додавати нові функції не углибляючись в скриптові мови. Веб-сайт має можливість повністю взаємодіяти з базою даних,він має: реєстрацію, авторизацію, різні права для захисту бази даних від несанкціонованих дій.  
Звичайно використовувати таку інформаційну систему цілком можливо, але для повної роботи інтернет-магазину потрібно, що б з базою даних взаємодіяли не тільки через веб-сайт, але і через окрему інформаційну систему складу з товарами, тому що тільки так можливо чітко знати, які товари є на складі і в якій кількості.  
Розроблена в даній курсовій роботі база даних і веб-сайт інтернет-магазину в цілому можна вважати краще за свої аналоги.   
База даних була розроблена на СУБД mySQL, яка має властивості: швидкість роботи, масштабованість і безпеку. Завдяки чому зв'язок з базою даних завжди буде на високому рівні, проект завжди можна розвивати далі, робити його більше, додавати нові можливості, таблиці, запити(що дуже важливо при розробці саме інтернет-магазинів) і при всьому цьому база даних завжди буде під захистом.  
Веб-сайт був розроблений з максимальною простотою, завдяки чому: користувач буде бачити тільки найважливіше, сайт буде працювати дуже швидко. Це є дуже важливим, тому що головна характеристика сайту це і є простота його використання і швидкість роботи. Користувач який буде мати гарні враження від використання веб-сайту, з більшим шансом звернеться до його послуг в наступний раз!  
Курсовий проект можна вважати успішно виконаним, тому що всі запроектовані завдання було виконано.

**ЛІТЕРАТУРА**

1. Основи проектування баз даних: Навчальний посібник. − К.: КНУБА, 2005. – 204 с.
2. SQLite vs MySQL vs PostgreSQL: сравнение систем управления базами даннях [Електроний ресурс]. Доступ за посиланням: [http://devacademy.ru/posts/sqlite-vs-mysql-vs-postgresql/](http://devacademy.ru/posts/sqlite-vs-mysql-vs-postgresql/%20).[Дата звернення 29.05.2018 р.]
3. Angular vs. React vs. Vue: Сравнение 2017 [Електроний ресурс]. Доступ за посиланням: <https://habr.com/post/338068/>. [Дата звернення 29.05.2018 р.]
4. 5 причин використовувати Монго (MongoDB) [Електроний ресурс]. Доступ за посиланням: <http://echo.lviv.ua/dev/9693>. [Дата звернення 29.05.2018 р.]
5. Microsoft SQL Server wikipedia [Електроний ресурс]. Доступ за посиланням: <https://uk.wikipedia.org/wiki/Microsoft_SQL_Server>. [Дата звернення 29.05.2018 р.]
6. MongoDB wikipedia [Електроний ресурс]. Доступ за посиланням: <https://uk.wikipedia.org/wiki/MongoDB>. [Дата звернення 29.05.2018 р.]
7. MySQL и MongoDB — когда и что лучше использовать [Електроний ресурс]. Доступ за посиланням: <https://habr.com/post/322532/>. [Дата звернення 29.05.2018 р.]
8. Інформаційні процеси wikipedia [Електроний ресурс]. Доступ за посиланням: <https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D1%96%D0%B9%D0%BD%D1%96_%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D0%B8>. [Дата звернення 29.05.2018 р.]
9. Creatly конструювання uml і інших діаграм [Електроний ресурс]. Доступ за посиланням <https://creately.com/>. [Дата звернення 29.05.2018 р.]
10. TypeScript wikipedia [Електроний ресурс]. Доступ за посиланням: <https://uk.wikipedia.org/wiki/TypeScript>. [Дата звернення 29.05.2018 р.]
11. Bootstrap wikipedia [Електроний ресурс]. Доступ за посиланням: <https://ru.wikipedia.org/wiki/Bootstrap_(%D1%84%D1%80%D0%B5%D0%B9%D0%BC%D0%B2%D0%BE%D1%80%D0%BA)>. [Дата звернення 29.05.2018 р.]
12. PHP wikipedia [Електроний ресурс]. Доступ за посиланням: <https://uk.wikipedia.org/wiki/PHP>. [Дата звернення 29.05.2018 р.]
13. WebStorage wikipedia [Електроний ресурс]. Доступ за посиланням: <https://uk.wikipedia.org/wiki/Web_Storage>. [Дата звернення 29.05.2018 р.]
14. JSON wikipedia[Електроний ресурс]. Доступ за посиланням: <https://uk.wikipedia.org/wiki/JSON>. [Дата звернення 29.05.2018 р.]
15. AJAXwikipedia [Електроний ресурс]. Доступ за посиланням: <https://uk.wikipedia.org/wiki/AJAX>. [Дата звернення 29.05.2018 р.]

**ДОДАТКИ**

На перетині рядків і стовпців зазначено дії, які може виконувати користувач даного типу:

**1** – перегляд даних, **2** – редагування, **3** – видалення, **4** – повний доступ.

Матриця доступу для ролі «Покупці»

|  |  |  |  |
| --- | --- | --- | --- |
|  | Товари | Замовлення | Профіль |
| Відвідувач | 1 | 0 | 0 |
| Користувач | 1 | 1 | 2 |

**Таблиця Б1.**

Матриця доступу для ролі «Керівники»

|  |  |  |  |
| --- | --- | --- | --- |
|  | Товари | Замовлення | Профіль |
| Адміністратор | 1 | 1 | 4 |
| Модератор | 4 | 1 | 4 |
| Оператор | 1 | 3 | 2 |

**Таблиця Б2.**