ОДЕСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ І. І. МЕЧНИКОВА

(повне найменування закладу вищої освіти)

Факультет математики, фізики та інформаційних технологій

(повне найменування факультету)

Кафедра інформаційних технологій

(повна назва кафедри)

Кваліфікаційна робота

на здобуття ступеня вищої освіти «Бакалавр»

«Розробка веб-сайту на базі платформи ASP.NET Core»

(тема кваліфікаційної роботи українською мовою)

«Development of a Website Using ASP.NET Core Platform»

(тема кваліфікаційної роботи англійською мовою)

Виконав: здобувач денної форми навчання

спеціальності 122 Комп'ютерні науки

(код, назва спеціальності)

Освітня програма Комп'ютерні науки

(назва)

Якименко Андрій Русланович

(прізвище, ім'я, по-батькові здобувача)

Керівник к.т.н., доцент Фразе-Фразенко О.О. _____

(науковий ступінь, вчене звання, прізвище, ініціали) (підпис)

Рецензент к.т.н., доцент Щербіна Ю.В.

(науковий ступінь, вчене звання, прізвище, ініціали)

Рекомендовано до захисту:
Протокол засідання кафедри
Інформаційних технологій .
№ 1 від 09 червня 2024 р.

Завідувачка кафедри

_____ КАЗАКОВА Надія .

(підпис) (прізвище,ім'я)

Захищено на засіданні ЕК № 13 .
протокол № 31 від 21 червня 2024 р.

Оцінка добре / С / 80 .

(за національною шкалою/шкалою ECTS/ бали)

Голова ЕК

_____ КОПИЧЕНКО Іван .

(підпис) (прізвище,ім'я)

Одеса 2024

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

MVC – Model-view-controller– паттерн у розробці сайтів.

ІС – інформаційна система.

GPL – GNU General Public License – загальна публічна ліцензія GNU.

Хостинг – послуга з надання простору для розміщення сайтів в Інтернеті.

Apache – вільний WEB-сервер.

CSS – Cascading Style Sheets – каскадні таблиці стилів.

HTML – HyperText Markup Language – мова гіпертекстової розмітки.

MySQL – вільна система управління базами даних (СУБД).

JS- Java Script – мова написання сценаріїв.

ODBC – OpenDatabaseConnectivityStandard – стандарт підключення до відкритих баз даних.

SADT – Structured Analysis and Design Technique – методологія структурного аналізу і проектування.

UML – Unified Modeling Language – мова візуального моделювання.

URI – Uniform Resource Identifier – уніфікований ідентифікатор ресурсу.

ВСТУП

На сьогоднішній день інформаційні системи переплелися з усіма аспектами нашого життя, вони проникли в сфери роботи, розваг, освіти, комунікацій та інші. Без їхньої присутності важко уявити будь-яку діяльність, від управління підприємством та іншого подібного, до навчання в університеті або звичайного спілкування з друзями. До того ж сучасні організації активно використовують інформаційні системи для оптимізації своїх процесів та забезпечення більшої конкурентоспроможності. Вони використовують широкий спектр програмних продуктів для автоматизації бізнес-операцій, ведення обліку, аналізу даних та прийняття стратегічних рішень. Саме такий дієвий підхід дозволяє підприємствам надати ефективного управління своїми ресурсами, швидко реагувати на ринкові зміни умов, адаптуватися до можливостей продавців і досягати успіху в умовах постійної конкуренції.

В такому світі, де в кожному куточку нашого життя міститься все більше даних, забезпечення очищення та злиття їх у виробничі та бізнес-процеси стає вищою метою. Фактично у будь-якій галузі господарювання, від виробництва до роздрібної торгівлі, інформаційні системи є критичними для успішності та життєздатності. Наприклад, давайте розглянемо потік поставки коробок [1]. Це може вважатися незначним елементом найбільшого ланцюга поставки. Перш ніж зробити такий висновок, слід визнати, що це є дуже складним процесом для врахування на дуже багатьох рівнях. Для прикладу: ви як виробник коробок для перевезення товарів. Тут потрібно буде вести облік не тільки коробок, що є на складі, а й ті, що приходять та відправляються, а так само слідкувати за їхнім станом та умови транспортування.

Практичну реалізацію поставлених цілей повинна врешті-решт розрішувати розроблена інформаційна система. Вона має дати можливість проводити облік коробок за допомогою зручного інтерфейсу, автоматизувати процеси їх отримання та відображати аналітичні звіти щодо ефективності саме поставок та управління ресурсами. Однак це, скоріш за все, буде лише один з

частково розв’язаних у будь-якій нормальній програмі інформаційної системи великого пазла. Інформаційна система для управління поставками ділить простір з іншою класом інформаційні системи – системами обліку на слідкові системи та бухгалтерським управлінням. Тому, вона також повинна інтегрувати в один великорозмірний ланцюг, як зазначалося, інформацією функціональні навчальні програми на підпадінг з відстеженням. Таким чином, ця дипломна робота має відповідне вирішення такої ідеї.

У рамках дипломної роботи було проведено ретельне дослідження та аналіз сучасних підходів до управління поставками, розроблено та реалізовано веб-ресурс для автоматизації цих процесів.

Диплом містить 60 сторінок, 23 рисунки, 3 таблиці, 11 посилань.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ І ПОСТАНОВКА ЗАВДАННЯ

1.1 Дослідження предмета та особливостей мережевого WEB-ресурсу

Не завжди можна якісно та точно відстежити усі зміни та усі нюанси у складській діяльності. Такі речі, як відстеження стану товару, його кількості, а також від якої поставки прийшов брак – це завжди дуже складно, а також якщо брати у увагу так званий «людський фактор» не можна виключити і достатньо багато помилок, які можуть нанести збитки, а також навіть зруйнувати бізнес як слідство. В свою чергу навіть самі різноманітні цифрові технології позбавленні цього самого «людського фактору», а значить не схильні до помилок або любих інших прояв людської неуважності, до забутьковатості.

Інформаційні технології є ключовим фактором для оптимізації логістичних процесів та управління поставками. Застосування систем управління запасами (WMS), систем відстеження поставок (TMS) та інших інноваційних рішень може значно підвищити ефективність бізнесу та знизити витрати. Важливо визначити найсучасніші технологічні рішення, які найкраще відповідають потребам конкретного сегменту ринку.

Розвиток технологій та впровадження карантинних обмежень підштовхує бізнес до переходу в онлайн-середовище. Це створює можливості для ефективної взаємодії між підприємствами та споживачами, проте також поставляє нові виклики, пов'язані з організацією робочих процесів та забезпеченням комфорту та зручності для користувачів. Зокрема, потрібно зауважити, що більш вікове ком’юніті не полюбляє складні та багаторівневі інтерфейси взаємодії з програмою, тому потрібно розробити інтерфейс, що буде простим, без зайвих елементів та складових. Відповідно таким чином буде досягнено ширше охоплення аудиторії

При розробці мережевого WEB-ресурсу для обліку поставок коробок, необхідно враховувати не лише технічні аспекти, але й психологічні та філософські відтінки взаємодії з користувачами. Важливо створити середовище,

яке сприятиме ефективному обміну інформацією, а також надасть можливість відчути зв'язок з реальним світом, навіть у віртуальному просторі.

У сучасній епохі цифрових можливостей і воєнного конфлікту люди мають потребу в швидкому та безпечному доступі до цифрових сервісів, уникненні зайвих труднощів і контактів. Внаслідок військового стану в країні значна частина торгівельних точок та підприємств переходить до онлайн-формату, навіть у таких сферах, як облік поставок. Але електрона форма ведення дає змогу дати не тільки більш детальний та точний погляд, що досить важко зробити у фізичному вигляді за допомогою паперових зразків, вона ще й мінімізує кількість паперової роботи та економить час користувача.

1.2 Огляд форматів зображень для WEB-розробки, огляд видів верстки WEB-сторінок

Існують десятки та навіть сотні графічних форматів, але тільки деякі з них підтримуються WEB-браузерами.

Деякі з характеристик графічних файлів:

– Прозорість: Ця характеристика визначає, наскільки прозорий може бути фон або окремі частини зображення. Прозорість дозволяє зображенню взаємодіяти з тлом або іншими об'єктами на веб-сторінці, що робить його більш адаптивним і естетичним.

– Стиснення: Це процес зменшення розміру файлу зображення без або з втратами якості. Різні методи стиснення можуть бути використані для зменшення обсягу даних, що робить файли зображень меншими і швидше завантажуваними.

– Кольорова гама: Ця характеристика визначає кількість кольорів, які можуть бути використані в зображенні. Вона може бути обмеженою (наприклад, 256 кольорів у форматі GIF) або без обмежень (наприклад, 24-бітна кольорова гама у форматі JPEG).

– Роздільна здатність: Це кількість пікселів, які складають зображення. Вища роздільна здатність дозволяє зображенню бути більш деталізованим і якісним, але також може призвести до більшого розміру файлу.

– Анімація: Деякі графічні формати дозволяють створювати анімовані зображення, що складаються з послідовних кадрів. Це відкриває можливості для створення цікавих та виразних веб-елементів.

Визначимо графічні формати, які підтримують більшість браузерів та їх переваги:

1. JPEG (Joint Photographic Experts Group) – це один з найбільш поширених форматів зображень із високим рівнем стиснення. Він підтримується всіми браузерами. З переваг можна відмітити 24-бітний колір, що робить його привабливим варіантом для розміщення на сайті. Велике стиснення дозволяє швидко завантажувати його в браузері при мінімальних втратах якості.
2. PNG (Portable Network Graphics) - PNG також підтримує 24-бітний колір, а також прозорість, що робить його ідеальним при роботі з альфа каналом. З переваг можна відмітити високу якість зображення, підтримку редагування прозорості.
3. GIF (Graphics Interchange Format) – це формат для анімованих зображень, але підтримує всього 256 кольорів. Підтримує прозорість. З переваг можна відмітити можливість створювати анімації на сайті без додаткових плагінів та інструментів.
4. SVG (Scalable Vector Graphics) – Це векторний формат зображення, що використовує математику, що дозволяє масштабувати зображення без втрат якості взагалі. З переваг можна відмітити те, що цей формат підходить для розміщення будь яких зображень без втрат якості, тому що кожна лінія на зображені будується завдяки математичним операціям, тому цей формат підходить для усіх роздільних здатностей, дрібних логотипів, а також багатьох інших графічних елементів на сайті.

5. ICO (Icon) – спеціально призначений для іконок формат зображення. З переваг можна відмітити те, що це найбільш привабливий формат зображення для розміщення піктограм на веб-сторінках та веб-додатках.

   Всі ці формати можуть бути використані на сайті для розміщення контент. В залежності від різних факторів та цілей можна підібрати той формат, що якомога більше буде підходити під конкретну задачу.

Перейдемо до типів верстки[8]. Кожен тип верстки має свої унікальні характеристики, переваги та недоліки. Вибір відповідного виду верстки залежить від конкретних потреб проекту, цільової аудиторії та типу контенту. Наприклад, для сучасних веб-сайтів, орієнтованих на широку аудиторію з різними пристроями, найбільш підходящою буде чуйна верстка. Для проектів, де важливо мати повний контроль над дизайном на різних розмірах екранів, адаптивна верстка може бути кращим вибором.

Розглянемо основні види верстки, їх плюси та мінуси:

1. Фіксована верстка – це найбільш простий варіант верстки. Всі розміри задаються у пікселях тому всі розміри залишаються статичними незалежно від роздільної здатності та розмірів вікна браузера. З плюсів можна відмітити повний контроль над розташуванням і розмірами елементів, а також простоту у розробці, тому що таку верстку легше реалізувати і передбачити кінцевий вигляд. З мінусів можна відмітити відсутність адаптивності, особливо на мобільних пристроях.
2. Резинова верстка – розміри встановлюються у відсотках або у vh та vw одиницях, а це в свою чергу дозволяє елементам на сторінці масштабуватися відповідно до розміру вікна браузера. Така верстка найбільш підходить для сайтів з динамічним контентом, який повинен змінюватись в залежності від розміру вікна браузера. З переваг можна відмітити гнучкість, а саме те що елементи будуть непогано адаптуватися під різні роздільні здатності і розміри вікна браузера, позіціонуватися гнучко зберігаючи пропорції. З недоліків можна

відмітити те, що передбачити кінцевий вигляд буде більш складно, чим у варіанті с фіксованою версткою, а також цей варіант є більш складним у реалізації та тестуванні.

3. Адаптивна верстка – це найбільш поширений варіант комерційної верстки. Використовує фіксовані макети під різні типи пристроїв (екрани моніторів, екрани ноутбуків, екрани планшетів, екрани мобільних телефонів).

1.3 Огляд та аналіз існуючих систем

Для успішної розробки та реалізації ІС, яка буде забезпечувати стабільний облік поставок, їх моніторинг стану, а також відстеження походження було прийнято рішення спочатку визначити аналоги, які допоможуть визначити вимоги до цієї системи та функції, які вона повинна виконувати заради задоволення бізнес потреб користувачів. Одним з найважливіших етапів цього процесу є підпроцес аналізу існуючих аналогів, що дозволить зрозуміти актуальні тенденції у цьому напрямку, а також якомога більш корисні практики.

Аналізуючи функціональні можливості та архітектурні рішення вже існуючих систем, можна отримати цінну інформацію про переваги та недоліки різних підходів до розробки подібних ресурсів. Це допоможе не лише уникнути повторення помилок, які були допущені іншими розробниками, але й розробити ефективні рішення, які вже довели свою працездатність на практиці.

На сучасному ринку існує мала кількість веб-ресурсів, які надають послуги з обліку товарів і моніторингу за товарами. Деякі з них спеціалізуються на управлінні логістичними процесами, а інші відстежують стан товарів в режимі реального часу. Важливо зрозуміти, які функції та інструменти є найбільш корисними для кінцевих користувачів і як їх можна адаптувати до конкретних вимог проекту.

У цьому розділі ми розглянемо різні існуючі системи, які пропонують подібну функціональність. Ми проводимо детальний аналіз їх можливостей, архітектурних рішень, інтерфейсів і користувацького досвіду.

Знайти велику кількість WEB-ресурсів які були б аналогом розробляємого ресурсу не вдалося.

Доглянувши і проаналізувавши подібні систему, я зміг з'ясувати основні переваги і недоліки таких систем, на основі яких вдалося сформулювати вимоги до розробляємого WEB-ресурсу.

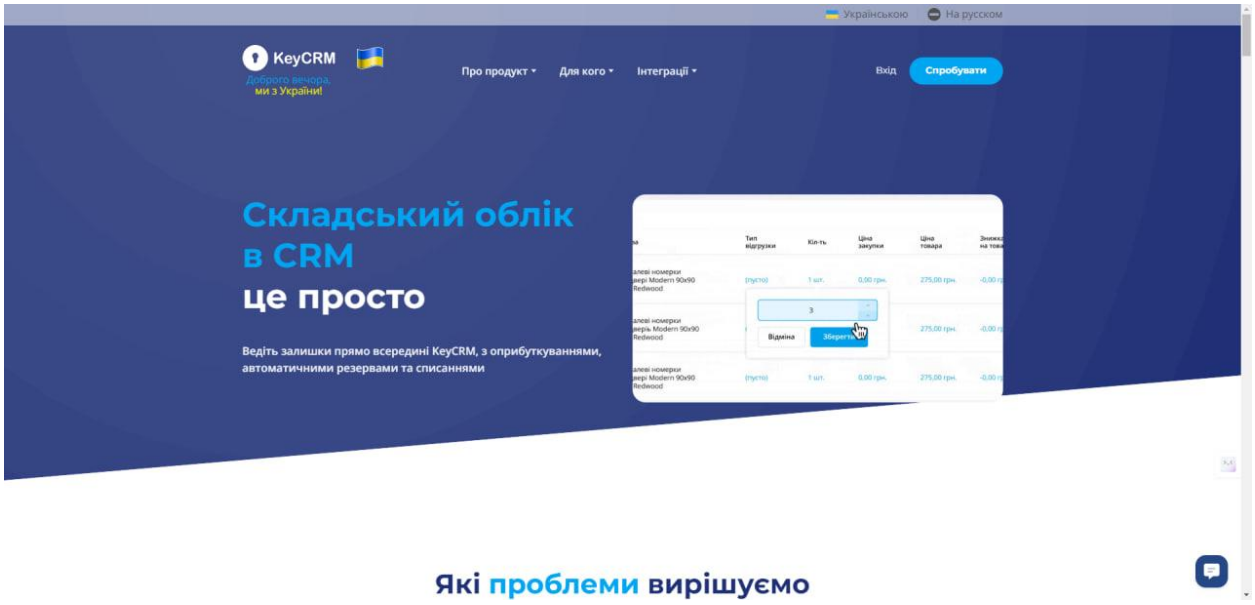

Рисунок 1.1 – Головна сторінка сайту «KeyCrm»

На головній сторінці сайту Keycrm [2] є дуже багато непотрібної для безпосередньої роботи інформації. Зокрема потрібно зауважити, що для того щоб почати роботу потрібно зробити свій обліковий запис, а також купити підписку, яка є не дешевою. Безкоштовна версія більш спрямована тільки на ознайомлення, а не на безпосередню роботу. Також у сервісі можна знайти посилання на завантаження мобільної версії системи, яка нажаль не працює. Система на телефонній версії навіть не може авторизувати у обліковий запис.

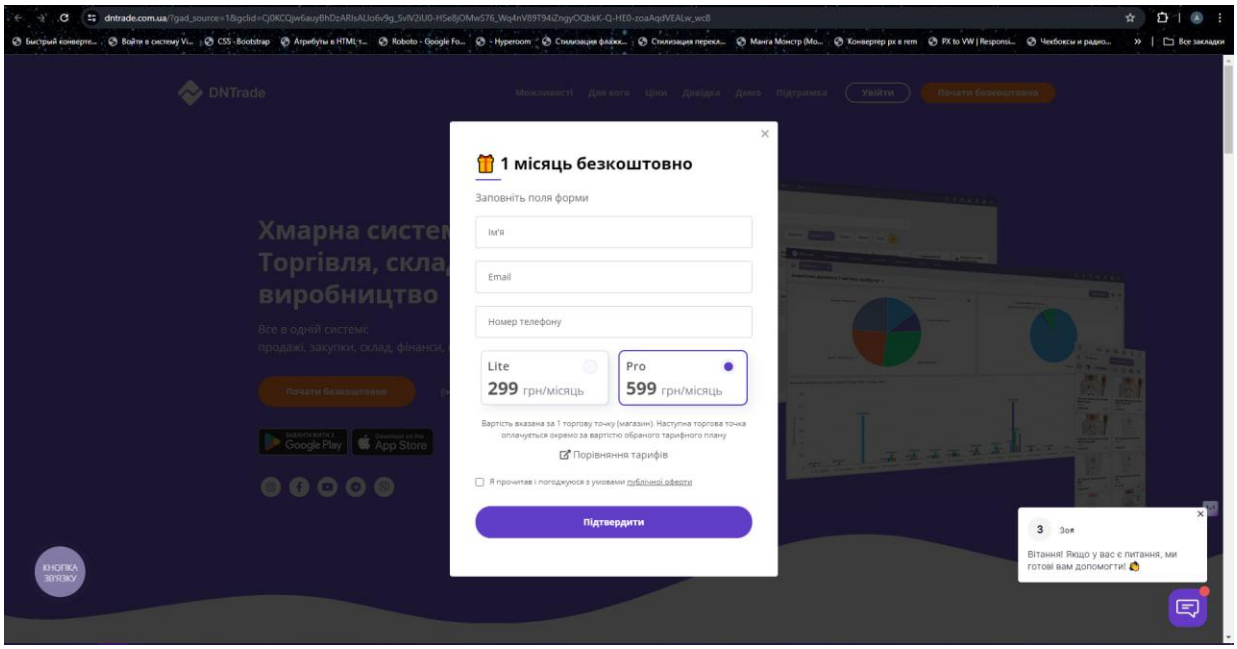

Рисунок 1.2 – Головна сторінка сайту «DNtrade»

На головній сторінці сайту DNtrade[3] нам одразу пропонують купити підписку, а також пропонують місяць безкоштовно спробувати сервіс. Гроші, звісно ж, списуються одразу, це було перевірено через пусту картку банку. Також на сайті є інтеграції з великою кількістю сторонніх сервісів, що нажаль ставить під питання безпеку персональних даних користувачів. Сайт також надає можливість використовувати підключення сканерів, принтерів чеків та цінників, банківських терміналів, ваг з прямим підключенням та з друком етикетки, моніторів клієнта. Робота з терміналами збору даних.

Цей сервіс [4] дозволяє не тільки вести учет поставок, а й комплектувати замовлення, контролювати прибуття товару, а також вести адресне зберігання, що на мою думку, як на думку розробника, є зайвим. Також цією програмою неможливо користуватись безкоштовно. Ціни на ліцензію, яка повинна бути купленою для використання ПЗ лякають. Цінники починаються від 10000 гривень і закінчуються цінником в 30000 гривень за одну ліцензію. Зокрема треба зауважити, що програма ніби гарантує прискорення роботи складу у 10 разів, що на мою думку є недостовірною інформацію, тому як прискорення

роботи у 10 разів потребую виключного ПЗ, яке буде повністю автоматизувати усі процеси, що майже неможливо.

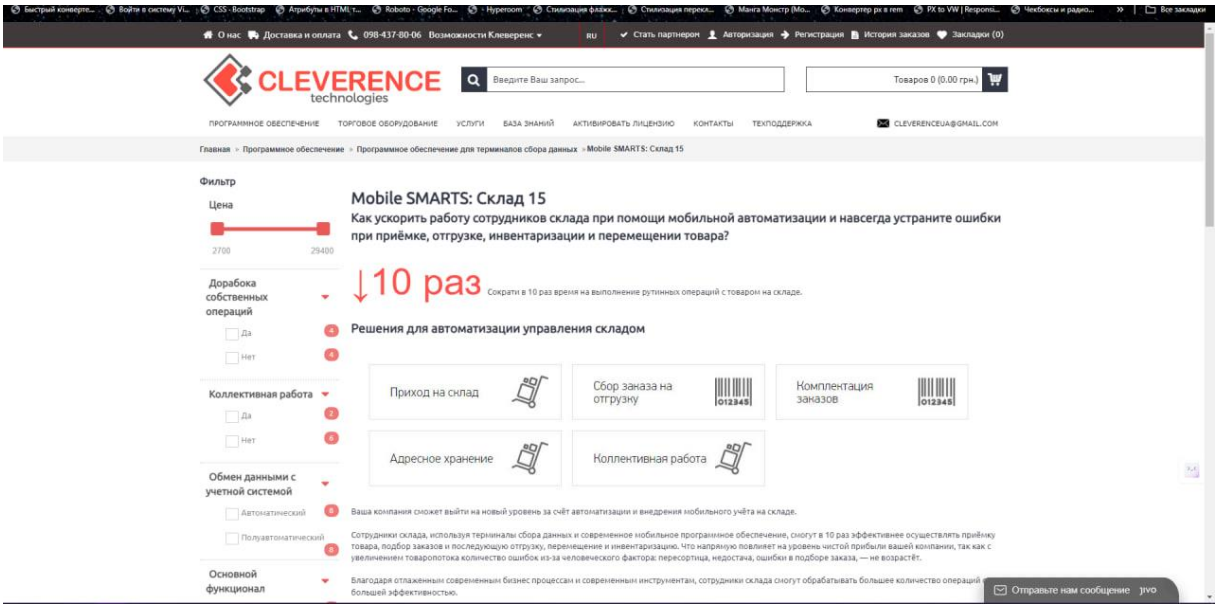

Рисунок 1.3 – Головна сторінка сайту «Cleverence»

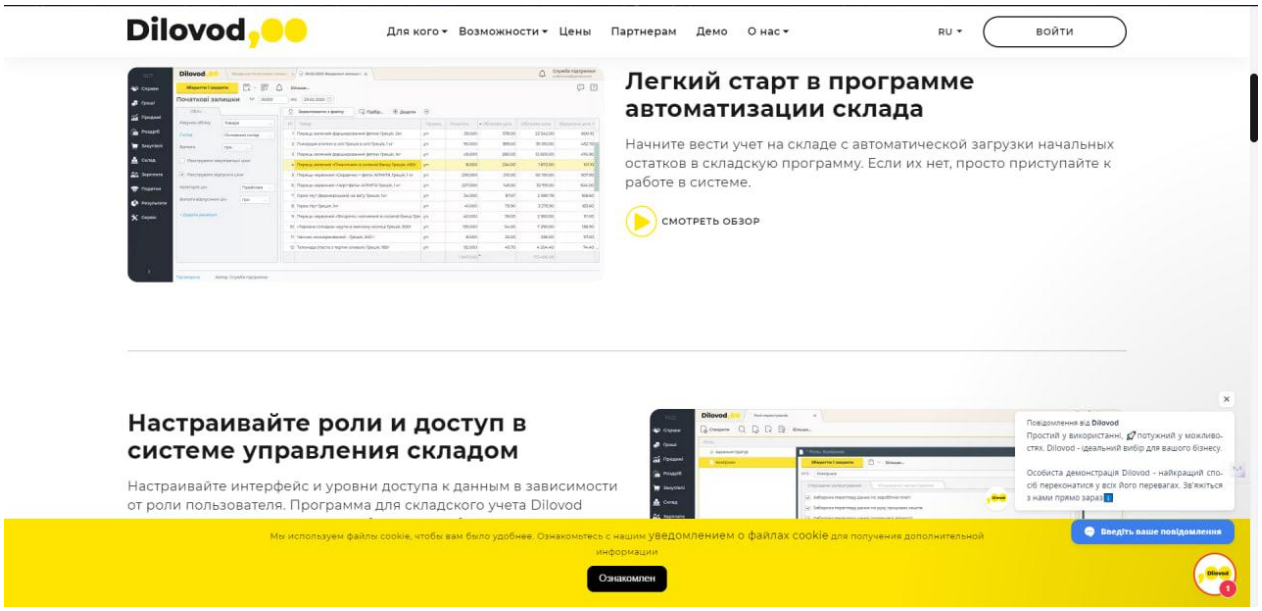

Рисунок 1.4 – Головна сторінка сайту «Dilovod»

Сайт [5] пропонує просту систему електронного учету продуктів. Нажаль, не підтримує стан поставок, звідки та куди вони йдуть, а також компанія

пропонує одразу купити в них підписку. Сама таблиця виглядає дуже просто, майже як у программі ексель.

Усі ресурси були випробувані. Дизайн сторінок виглядає занадто нагромадженним, орієнтуватися в інтерфейсі дуже складно. У деяких системах оновлення проходять не завжди, а також є багато лімітів на використання, особливо у безкоштовних версіях.

1.4 Постановка завдання

Метою створення мережевого WEB-ресурсу для учету поставок є забезпечення інформаційної присутності компанії в мережі Інтернет та надання послуг фірми якомога більшому числу потенційних споживачів.

Користувач повинен бути задоволений усіма аспектами роботи у сервісі, тому потрібно розробити простий та зрозумілий UI/UX дизайн, що задовільнить користувача. Зокрема користувач повинен мати знання про політику конфіденційності сервісу задля уникнення конфліктних ситуацій. Розробка інформаційної системи для обліку поставок коробок, їхнього стану та походження є важливим етапом для оптимізації логістичних процесів будь-якої компанії. Основною метою даного проекту є створення веб-ресурсу, який дозволить користувачам зручно та ефективно управляти всіма аспектами постачання товарів. Враховуючи актуальні тенденції в галузі дизайну та користувацького досвіду, система повинна бути простою, інтуїтивно зрозумілою та ненагромадженою. Визначимо пункти постановки завдання:

1. Основні функції:

– Система обліку. Користувачі повинні мати можливість реєструвати нові поставки, вносити інформацію про місце відправки, вказувати кількість та тип коробок. Все це повинно бути не тільки для ручного внесення, а й для автоматичного задля економії часу;
– Моніторинг стану коробок. Система повинна дозволяти відстежувати стан кожної коробки в режимі реального часу, статус пошкоджень.

– Відстеження походження. Важливо, щоб користувачі могли бачити історію переміщень кожної коробки, включаючи пункти відправлення, дати та час;
– Дві сторінки з таблицями для кожної коробки. В таблиці повинен бути номер поставки, номер коробки, унікальний ідентифікатор, який буде генеруватися сам, а також стан коробки та кнопку для редагування.

2. Дизайн:

– Інтерфейс повинен бути максимально простим та інтуїтивно зрозумілим, щоб навіть користувачі без спеціальної підготовки могли легко взаємодіяти із системою;
– Використання сучасних принципів UX/UI дизайну для забезпечення позитивного користувацького досвіду;
– Інтерфейс повинен бути чистим, з мінімальною кількістю елементів на кожній сторінці, щоб уникнути перевантаження користувача інформацією;
– Використання зрозумілих іконок та чітких підписів для навігаційних елементів;
– Логічне розташування елементів інтерфейсу, що дозволяє швидко знайти потрібну функцію або інформацію;
– Мінімізація кількості кліків для виконання основних операцій;
– Дизайн повинен бути резиновим, щоб забезпечити зручне користування на різних типах екранів;
– Використання єдиного стилю у всьому інтерфейсі для створення узгодженого та професійного вигляду.

Успішне виконання цього завдання вимагатиме ретельного планування, аналізу вимог та постійного тестування системи для забезпечення її відповідності потребам користувачів та ефективності у повсякденному використанні.

2 ВИБІР АРХІТЕКТУРИ СИСТЕМИ ТА ПРОГРАМНИХ ЗАСОБІВ РЕАЛІЗАЦІЇ

2.1 Вибір платформи

Платформою для реалізації завдання було обрано платформу ASP.NET.

ASP.NET — це потужний фреймворк для розробки веб-додатків, створений корпорацією Microsoft. Він є частиною більшої платформи .NET і надає розробникам всі необхідні інструменти для створення динамічних веб-сайтів, веб-сервісів та інших інтернет-додатків.[9]

Основні характеристики ASP.NET:

Серверна платформа: ASP.NET працює на сервері, обробляючи запити від клієнтів (веб-браузерів) і генеруючи відповідний HTML, який потім відправляється назад до клієнтів. Це дозволяє створювати динамічний контент, який може змінюватися в залежності від дій користувача або даних з бази даних.

Мови програмування: ASP.NET підтримує кілька мов програмування, включаючи C# і JavaScript[6], що надає гнучкість розробникам у виборі мови, з якою вони бажають працювати.

Вбудована безпека: Фреймворк[7] має вбудовані засоби для забезпечення безпеки, включаючи аутентифікацію та авторизацію користувачів, захист від поширених веб-загроз, таких як XSS (Cross-Site Scripting) та CSRF (Cross-Site Request Forgery).

Продуктивність: Завдяки компіляції коду на сервері, кешуванню і оптимізації, ASP.NET забезпечує високу продуктивність веб-додатків, що дозволяє обробляти велику кількість запитів.

Інтеграція з .NET: Оскільки ASP.NET є частиною платформи .NET, розробники можуть використовувати всі бібліотеки та інструменти .NET, що спрощує розробку та інтеграцію з іншими системами.

2.2 Вибір патерну розробки

Для розробки веб-додатка, який призначений для обліку поставок коробок, їхнього стану та походження, було вирішено використовувати патерн Model-View-Controller (MVC)[10] (рис 2.1).

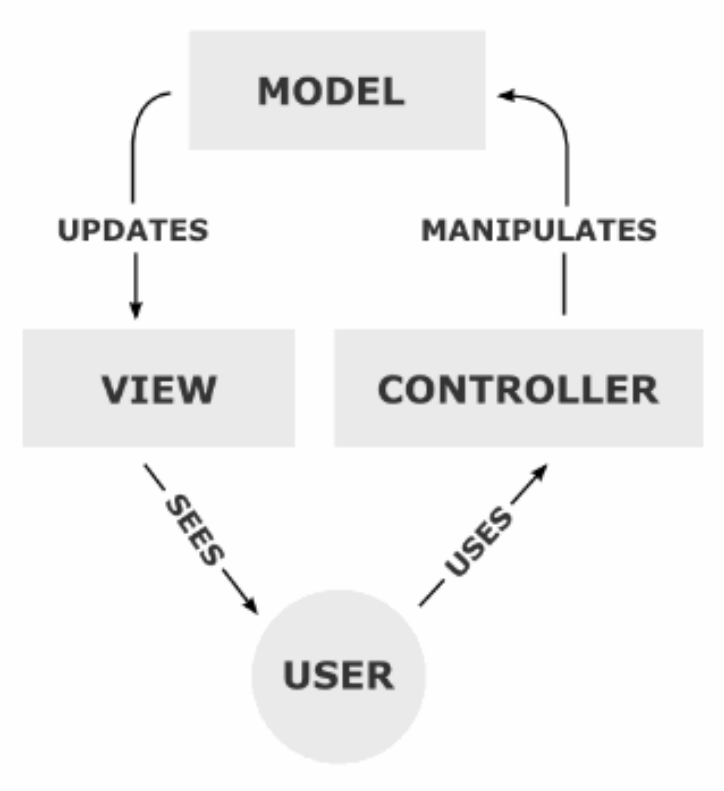

Рис 2.1 – Схема взаємодії патерну MVC

Цей вибір базується на кількох ключових причинах, які роблять MVC найбільш відповідним і ефективним для нашого проекту. Ця структура допомагає розділити бізнес-логіку додатка від його інтерфейсу користувача, що робить код більш організованим і легшим для підтримки. Модель відповідає за управління даними та бізнес-логікою, представлення — за відображення даних користувачу, а контролер — за обробку вхі-дних запитів від користувача, взаємодіючи з моделлю для отримання даних і з представленням для відображення результатів. Вибір патерну MVC для розробки веб-додатка

базується на кількох ключових перевагах, які роблять його ефективним і популярним серед розробників.

Перш за все, MVC забезпечує чітке розділення обов'язків між компонентами додатка. Це дозволяє розробникам працювати над окремими частинами незалежно один від одного, що підвищує ефективність команди та спрощує підтримку коду.

Ще однією важливою перевагою є повторне використання коду. Оскільки компоненти MVC добре ізольовані один від одного, їх легше повторно використовувати в різних частинах додатка або на-віть в інших проектах.

Наприклад, ту саму модель можна використовувати з різними представленнями, що значно зменшує кількість необхідного коду і підвищує ефективність розробки.

Полегшене тестування є ще одним вагомим аргументом на користь MVC. Патерн сприяє модульному підходу до тестування, оскільки моделі можна тестувати незалежно від представлень, а контролери — окремо від моделей. Це дозволяє швидше виявляти та виправляти помилки, підвищуючи надійність і якість програмного забезпечення.

Гнучкість та масштабованість — ще одні значні переваги MVC. Завдяки чіткій структурі, додатки на базі цього патерну легше масштабувати. Можна додавати нові функціональні можливості, не впливаючи на вже існуючий код. Це дозволяє адаптувати додаток до змін в бізнес-логіці або вимогах користувачів без значних витрат часу і ресурсів.

2.2 Вибір архітектури мережевого WEB-ресурсу

На основі проведеного огляду та аналізу систем-аналогів було встановлено, що для успішного функціонування мережевого веб-ресурсу компанії, яка надає послуги з фотозйомки та постобробки, необхідно розробити вдосконалену інформаційну веб-систему. Ця система повинна автоматизувати

збір, обробку та управління даними, використовуючи сучасні методи та веб-технології.

Мережевий веб-ресурс має включати базу даних, яка забезпечуватиме зберігання, швидкий доступ та пошук інформації. База даних об'єднує різнорідні дані та може організовувати їх за певними правилами. На сьогоднішній день веб-технології є найбільш перспективними для створення розподілених інформаційних систем з подальшою можливістю їх публікації в Інтернеті.

Клієнт-серверна архітектура інформаційної системи передбачає наявність двох взаємодіючих автономних процесів – клієнта та сервера, які, зазвичай, виконуються на різних комп'ютерах і обмінюються даними через мережу. Сервер баз даних виконує широкий спектр завдань з управління даними, забезпечуючи виконання наступних функцій [7]:

виконання призначених для користувача запитів на вибір і модифікацію даних і метаданих;

– зберігання і резервне копіювання даних;

– підтримка цілісності даних згідно з визначеними в базі даних правилам;

Робоче місце користувача може бути обладнане звичайним персональним комп'ютером, що дозволяє користувачеві зберегти звичне робоче середовище та комфорт під час виконання своїх завдань. Це означає, що користувачеві не потрібно вивчати нове програмне забезпечення або звикати до незнайомого інтерфейсу, оскільки всі необхідні інструменти вже доступні на його робочому комп'ютері.

Як архитектуру у проекті була обрана клієнт – серверна Архітектура(рис 2.2) У своїй основній конфігурації клієнт-серверна інформаційна система складається з таких ключових компонентів: перший це сервера баз даних, який відповідає за управління даними та обробку запитів від клієнтських додатків.

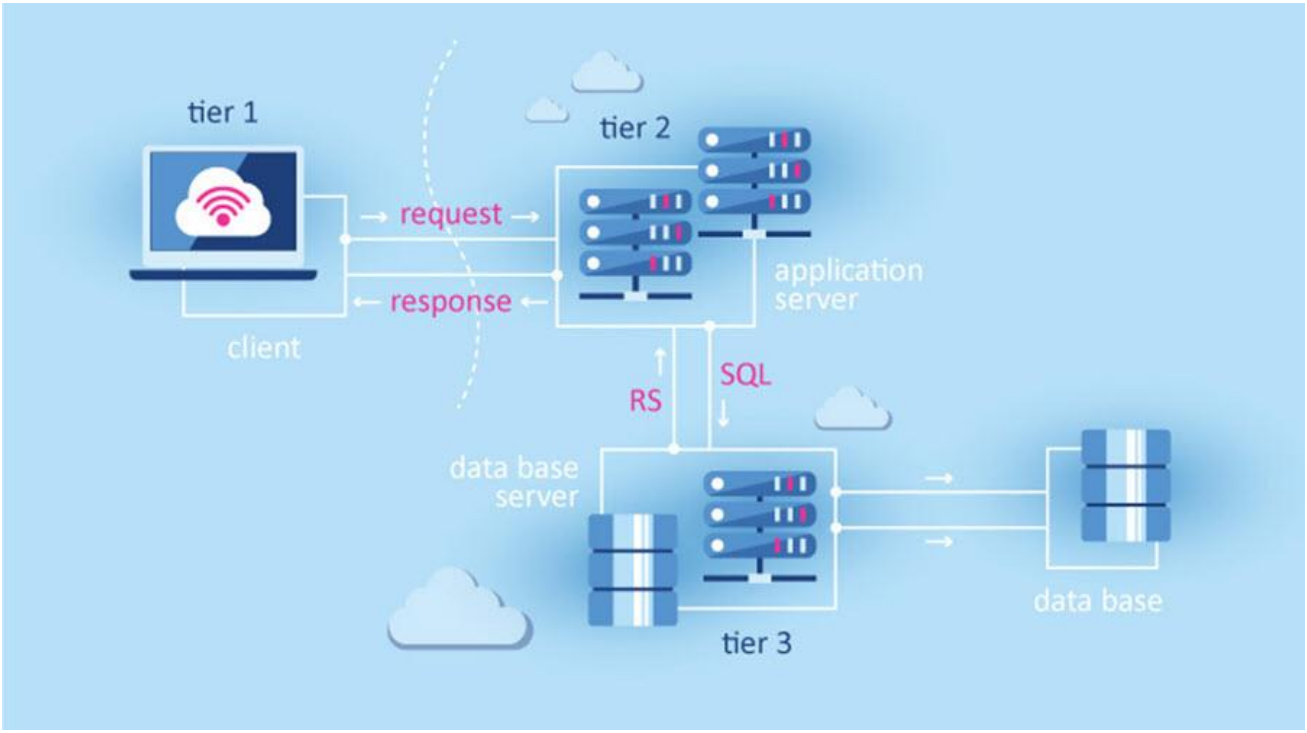

Рисунок 2.2 – Клієнт – серверна архітектура

Сервер баз даних виконує роль центрального сховища інформації, забезпечуючи цілісність та безпеку даних, а також ефективне їх зберігання і доступ клієнтських додатків, які надають користувачеві інтерфейс для взаємодії з системою та відправляють запити до сервера. Клієнтські додатки обробляють введені дані, відображають результати запитів та забезпечують зручність використання системи.

Окрім базової архітектури, існують і складніші реалізації клієнт-серверної архітектури. Однією з таких є багатоланкові інформаційні системи, які включають в себе сервери додатків. Сервери додатків виконують роль проміжного шару між клієнтами та сервером баз даних, реалізуючи бізнес-логіку і обробку даних. Це дозволяє розподілити навантаження на систему, підвищити її масштабованість та гнучкість, а також спрощує процес оновлення і підтримки програмного забезпечення.

Такі багатоланкові системи можуть містити декілька серверів додатків, що взаємодіють між собою та з іншими компонентами системи. Це дозволяє забезпечити високу продуктивність та надійність роботи системи, оскільки

завдання розподіляються між різними серверами, і у випадку відмови одного з них, інші сервери можуть продовжувати виконувати свої функції без переривань.

Таким чином, вибір клієнт-серверної архітектури для розробки інформаційної системи дозволяє досягти високої ефективності, надійності та зручності використання, що є ключовими факторами успішної роботи сучасних інформаційних систем.

2.3 Вибір системи управління базами даних

У XXI столітті використання систем управління базами даних (СУБД) є не просто технологічною необхідністю, а філософією, що відображає глибокі зміни в нашому суспільстві та підході до обробки інформації. Ми живемо в епоху, де інформація є новою валютою, а дані — серцем кожної організації. У цьому контексті СУБД стають фундаментом, на якому будується цифрова економіка.

Сучасний світ породив величезні обсяги даних, що постійно зростають, і ця інформація надходить з різних джерел: від соціальних мереж і мобільних додатків до інтернету речей і промислових сенсорів. Обробка таких масивів даних вручну стала б непосильним завданням, і саме тут на допомогу приходять СУБД, які дозволяють ефективно зберігати, управляти та аналізувати ці дані.

Крім технічної сторони, СУБД відображають філософію прозорості та підзвітності. Вони забезпечують структурованість і організованість даних, що дозволяє здійснювати контроль та аудит інформації. Це важливо не тільки для бізнесу, але й для суспільства в цілому, оскільки прозорість даних сприяє більшій довірі між громадянами та організаціями, урядами та бізнесами.

СУБД також підтримують ідею доступності знань. Вони дозволяють швидко знаходити необхідну інформацію та робити обґрунтовані рішення, базуючись на точних даних. Це особливо важливо в часи, коли швидкість і точність мають вирішальне значення. У медичній сфері, наприклад, СУБД допомагають лікарям швидко отримувати доступ до історії хвороби пацієнтів і приймати критично важливі рішення в найкоротші терміни.

Масштабованість СУБД відповідає нашому прагненню до зростання та розвитку. Вони легко адаптуються до змін, дозволяючи організаціям масштабувати свої системи у відповідь на зростання обсягів даних і потреб користувачів. Це відображає гнучкість та адаптивність, що є ключовими характеристиками успішних організацій у сучасному динамічному світі.

І, звичайно, не можна забувати про безпеку. У світі, де кібератаки та витоки даних стають все більш поширеними, СУБД надають надійні механізми захисту інформації. Вони гарантують, що дані залишаються конфіденційними, цілісними та доступними лише для авторизованих користувачів. Це підкреслює важливість безпеки в нашій цифровій реальності, де захист інформації є першочерговим завданням.

Отже, використання СУБД у 21-му столітті є не просто технологічним вибором, а необхідністю, яка відображає наші сучасні потреби та прагнення. Вони допомагають нам орієнтуватися в океані даних, робити обґрунтовані рішення, забезпечують прозорість та безпеку, і в кінцевому рахунку сприяють побудові ефективного та справедливого інформаційного суспільства.

Система управління базами даних (СУБД) є програмним забезпеченням, яке забезпечує створення, управління та обслуговування баз даних. Вона служить посередником між користувачами і базами даних, надаючи засоби для зберігання, маніпулювання та вилучення даних. СУБД забезпечує надійне зберігання даних у структурованому форматі, що дозволяє зберігати великі обсяги інформації та організовувати її таким чином, щоб забезпечити легкий доступ та ефективне керування. Вона надає можливість додавання, видалення та оновлення даних, організовуючи їх у вигляді таблиць, що містять рядки та стовпці, забезпечуючи логічну структуру для зберігання інформації. Користувачі можуть виконувати різноманітні операції з даними, такі як сортування, фільтрація, об’єднання та агрегування, що дозволяє отримувати потрібну інформацію з великих обсягів даних.

Підтримка транзакцій є однією з ключових функцій СУБД, що дозволяє виконувати групи операцій над даними як єдине ціле, забезпечуючи, що всі

операції виконуються успішно або жодна з них не виконується. Це критично важливо для забезпечення цілісності даних у випадку збоїв або помилок. СУБД також забезпечує паралельну обробку запитів, що дозволяє одночасно обслуговувати багато користувачів, забезпечуючи ефективний розподіл ресурсів та запобігання конфліктам.

СУБД надає інструменти для створення звітів та аналізу даних, що дозволяє користувачам витягувати корисну інформацію з бази даних та використовувати її для прийняття обґрунтованих рішень. Вона підтримує різні типи індексів, що дозволяє швидко знаходити потрібні дані, покращуючи продуктивність запитів. СУБД може працювати з різними типами даних, такими як текст, числа, дати та мультимедійні файли, забезпечуючи гнучкість у зберіганні та обробці різноманітної інформації.

У дипломній роботі використовується pgAdmin (рис 2.3) з кількох вагомих причин, які роблять його незамінним інструментом для роботи з базами даних PostgreSQL. PgAdmin є однією з найпопулярніших і найпотужніших графічних оболонок для управління PostgreSQL, що надає широкі можливості як для досвідчених адміністраторів баз даних, так і для новачків.

Однією з основних переваг pgAdmin є його зручний і інтуїтивно зрозумілий графічний інтерфейс. Це особливо важливо в контексті дипломної роботи, де часто потрібно візуально оцінювати структуру бази даних, створювати і модифікувати таблиці, переглядати і редагувати дані. PgAdmin дозволяє здійснювати всі ці дії за допомогою кількох кліків, що значно спрощує процес розробки та адміністрування.

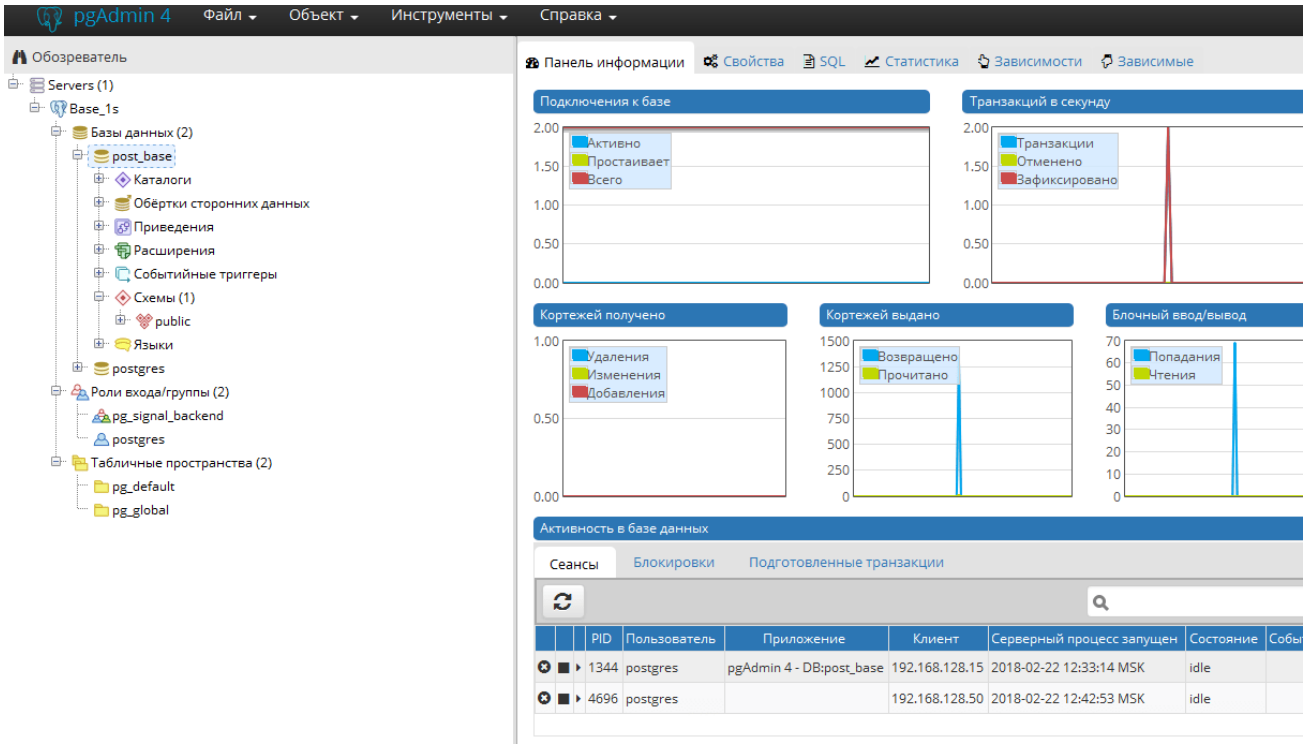

Рис 2.3 – Інтерфейс робочого вікна Pgadmin

Крім того, pgAdmin надає потужні інструменти для створення, налагодження та виконання SQL-запитів. Це дозволяє студенту швидко і ефективно тестувати різні SQL-запити, оптимізувати їх і аналізувати результати. Можливість зберігати часто використовувані запити і створювати складні скрипти робить pgAdmin незамінним інструментом для роботи з великими і складними базами даних.

Ще однією важливою перевагою є те, що pgAdmin підтримує роботу з декількома базами даних одночасно. Це дозволяє студенту легко керувати різними базами даних, переключатися між ними і порівнювати дані, що є важливим аспектом у багатьох дипломних проектах, де потрібно інтегрувати дані з різних джерел.

Безпека є ще однією сильною стороною pgAdmin. Він надає можливості для налаштування прав доступу до бази даних, створення користувачів і ролей з різними рівнями привілеїв. Це дозволяє забезпечити безпеку даних, контролювати доступ до важливої інформації і захищати базу даних від несанкціонованого доступу.

Що стосується конкретного вибору pgAdmin для цієї дипломної роботи, то важливо зазначити його сумісність і відкритість. PgAdmin є відкритим програмним забезпеченням з відкритим вихідним кодом, що робить його доступним для широкого кола користувачів. Це також означає, що інструмент постійно вдосконалюється і оновлюється завдяки активній спільноті розробників.

З точки зору інтеграції, pgAdmin легко інтегрується з іншими інструментами та технологіями, що використовуються в дипломному проекті. Це включає підтримку різних форматів даних, можливість експорту та імпорту даних, а також інтеграцію з системами контролю версій, такими як Git. Така гнучкість і сумісність роблять pgAdmin ідеальним вибором для сучасних проектів, де важлива інтеграція з іншими інструментами і системами.

Загалом, вибір pgAdmin у дипломній роботі обумовлений його зручністю, функціональністю, безпекою і сумісністю з іншими інструментами. Це робить його потужним і ефективним інструментом для управління базами даних, який допомагає досягти високих результатів у розробці та адмініструванні баз даних, забезпечуючи при цьому зручність і швидкість роботи.

2.4 Мова реалізації сценаріїв

Мовою реалізації сценаріїв було обрано JS. JavaScript був розроблений компанією Netscape Communications в середині 1990-х років. Спочатку він був створений Брендоном Айком під назвою Mocha, але незабаром був перейменований на LiveScript і, зрештою, на JavaScript. Метою створення мови було надання веб-сторінкам можливості виконувати невеликі сценарії безпосередньо в браузері, роблячи їх більш інтерактивними та динамічними.

Netscape Communications, відома своєю популярною серією браузерів Netscape Navigator, грала ключову роль у розвитку Інтернету. Вони розуміли необхідність мови сценаріїв, яка дозволяла б взаємодію з користувачем на веб-сторінках без необхідності постійного звернення до сервера. Ця концепція

призвела до створення JavaScript, який швидко став основною частиною веб-розробки.

У 1997 році JavaScript був стандартизований організацією ECMA International, що призвело до створення стандарту ECMAScript. Цей стандарт, відомий як ECMAScript, став основою для різних реалізацій JavaScript у браузерах. Сьогодні всі основні браузери підтримують сучасні стандарти ECMAScript, забезпечуючи сумісність і можливість використання нових функцій мови.

Щодо нормалізації, вона є важливим аспектом у розробці баз даних, який гарантує, що дані зберігаються ефективно і без надлишковості. Нормалізація – це процес організації даних у базі даних таким чином, щоб мінімізувати дублювання даних і забезпечити їх цілісність. Вона включає розподіл великих таблиць на менші і створення відносин між ними.

Існує кілька нормальних форм, які використовуються для нормалізації баз даних:

Перша нормальна форма (1NF) вимагає, щоб усі стовпці таблиці містили атомарні значення, тобто кожне значення в стовпці повинно бути неподільним.

Друга нормальна форма (2NF) вимагає, щоб таблиця знаходилася в 1NF, і всі неключові атрибути були функціонально залежними від усього первинного ключа.

Третя нормальна форма (3NF) вимагає, щоб таблиця знаходилася в 2NF, і жоден неключовий атрибут не був транзитивно залежним від первинного ключа.

Нормалізація допомагає уникнути проблем, таких як аномалії вставки, оновлення та видалення даних, що можуть виникати через надлишкові дані. Наприклад, без нормалізації зміна одного значення в таблиці може вимагати оновлення багатьох рядків, що підвищує ризик помилок і знижує ефективність. Нормалізація також покращує узгодженість даних, забезпечуючи, щоб кожен факт зберігався лише один раз. Проте, варто зазначити, що нормалізація може ускладнювати структуру бази даних, роблячи деякі запити більш складними і повільними через необхідність з'єднання багатьох таблиць. Тому в практичних

додатках часто використовується підхід денормалізації, де певні таблиці залишаються ненормалізованими для підвищення продуктивності запитів. Денормалізація часто використовується в системах, де швидкість читання даних має критичне значення.

Таким чином, компанія Netscape Communications створила JavaScript, який став невід'ємною частиною сучасного веб-розробки, а нормалізація баз даних забезпечує ефективне і надійне зберігання даних, мінімізуючи надлишковість і підвищуючи цілісність даних.

JavaScript має потужну екосистему, яка включає безліч бібліотек і фреймворків. Серед найбільш популярних бібліотек – jQuery, яка значно спрощує маніпуляції з DOM, подіями та анімаціями. Фреймворки, такі як React, Angular і Vue.js, надають структурований підхід до розробки складних додатків, полегшуючи управління станом, маршрутизацію та інші важливі аспекти розробки. Крім того, існує безліч інструментів для автоматизації завдань, тестування та збірки, таких як Webpack, Babel і ESLint, що робить розробку на JavaScript ще більш ефективною.

Одним з головних аргументів на користь JavaScript є його повсюдність. Майже всі сучасні веб-браузери підтримують JavaScript, що робить його незамінним інструментом для веб-розробки. Це також означає, що веб-додатки, написані на JavaScript, можуть працювати на різних платформах і пристроях, включаючи настільні комп'ютери, планшети і смартфони. Така широка підтримка гарантує, що ваші додатки будуть доступні для максимальної кількості користувачів. JavaScript також підтримує асинхронне програмування, що дозволяє розробникам створювати швидкі та чуйні додатки. Використовуючи колбеки, проміси та асинхронні функції, можна ефективно обробляти асинхронні операції, такі як запити до сервера, не блокуючи основний потік виконання програми. Це особливо важливо для веб-додатків, де швидкість і чуйність мають вирішальне значення для користувацького досвіду.

Порівнюючи JavaScript з іншими мовами програмування, варто згадати Python і Ruby, які також часто використовуються для веб-розробки. Python

славиться своєю простотою та читабельністю, а також потужними бібліотеками для наукових обчислень і аналізу даних. Однак, Python не підтримується нативно в браузерах, що обмежує його використання на клієнтській стороні. Ruby, з іншого боку, відомий завдяки фреймворку Ruby on Rails, який спрощує розробку веб-додатків. Проте, він менш поширений, ніж JavaScript, і має меншу екосистему.

JavaScript також виграє у порівнянні з мовами типу Java і C#. Хоча ці мови мають свої переваги, особливо у великих корпоративних проектах, вони зазвичай використовуються для серверної розробки і мають більш складний синтаксис. JavaScript, завдяки своїй простоті і гнучкості, дозволяє швидко прототипувати ітерації і забезпечує швидкий зворотний зв'язок, що є важливим у швидко змінюваному середовищі веб-розробки.

Крім того, JavaScript є мовою з динамічною типізацією, що дозволяє розробникам писати більш гнучкий код, хоча це може призводити до деяких помилок на етапі виконання. З появою TypeScript, надбудови над JavaScript з підтримкою статичної типізації, розробники отримали можливість користуватися перевагами обох підходів: динамічності JavaScript і безпеки статичної типізації.

У підсумку, вибір JavaScript для веб-розробки є очевидним завдяки його універсальності, потужній екосистемі, широкій підтримці в браузерах, можливостям асинхронного програмування і загальній популярності.

2.5 СУБД PostgreSQL

PostgreSQL — це об'єктно-реляційна система управління базами даних (ORDBMS), найбільш розвинена з відкритих СУБД у світі.

Однією з ключових привабливих рис PostgreSQL є його потужна продуктивність і можливість масштабування. Він легко впорається з великими обсягами даних, забезпечуючи високу швидкість виконання запитів, що робить його ідеальним вибором для проектів будь-якого розміру.

Крім того, PostgreSQL забезпечує широкий набір передових функцій та можливостей, включаючи підтримку різних типів даних, вбудовані процедури та тригери, а також багато іншого. Це робить його універсальним інструментом для вирішення найрізноманітніших бізнес-задач.

Крім усього іншого, PostgreSQL відомий своєю надійністю і стабільністю. Він регулярно оновлюється та підтримується спільнотою розробників, що гарантує його постійний розвиток та відповідність сучасним стандартам і вимогам.

Таким чином, PostgreSQL - це не лише база даних, але й потужний інструмент, який допомагає організаціям ефективно управляти своїми даними, розвиватися та досягати нових висот у своєму бізнесі.

2.6 Система контролю версій GitHub

GitHub — це платформа для хостингу та спільної розробки програмного забезпечення, заснована на системі контролю версій Git.

Основна філософія GitHub полягає у сприянні спільній роботі та відкритості. Завдяки можливості створювати репозиторії, розробники можуть зберігати весь свій код в одному місці, легко відслідковуючи історію змін та відновлюючи попередні версії при необхідності. Це особливо важливо в командній роботі, де одночасно над одним проектом можуть працювати десятки чи навіть сотні людей.

Однією з ключових функцій GitHub є пул реквести (pull requests). Вони дозволяють зберігати зміни для свого проекту на віддаленому репозиторії, а це означає, що доступ до свого проекта можна отримати з будь-якої точки земної кулі де є інтернет. Зокрема треба зауважити, що ця система дозволяє розробникам запропонувати свої зміни в коді, які потім можуть бути обговорені і переглянуті іншими учасниками проекту. Це створює середовище для конструктивної критики та покращення якості коду через колективне обговорення і аналіз. Окрім цього, GitHub інтегрується з різними інструментами

для автоматизації тестування і розгортання, що робить процес розробки більш безперебійним і ефективним.

У порівнянні з іншими системами, такими як GitLab або Bitbucket, то кожна з них має свої унікальні особливості. Наприклад, GitLab пропонує вбудовані CI/CD інструменти, які дозволяють автоматизувати процеси тестування і розгортання без необхідності інтеграції сторонніх сервісів. Bitbucket, з іншого боку, добре інтегрується з іншими продуктами Atlassian, такими як Jira, що може бути зручно для команд, які використовують ці інструменти, проте саме GitHub став своєрідним стандартом в індустрії.

Зокрема такі системи контролю версій як GitHub знамениті своєю системою диференціювання, а саме розділенням версій проекту на головну та побочні гілки (рис 2.4), що дозволяє розробнику ефективно та безпечно вести різноманітні експерименти без загрози для головної гілки.

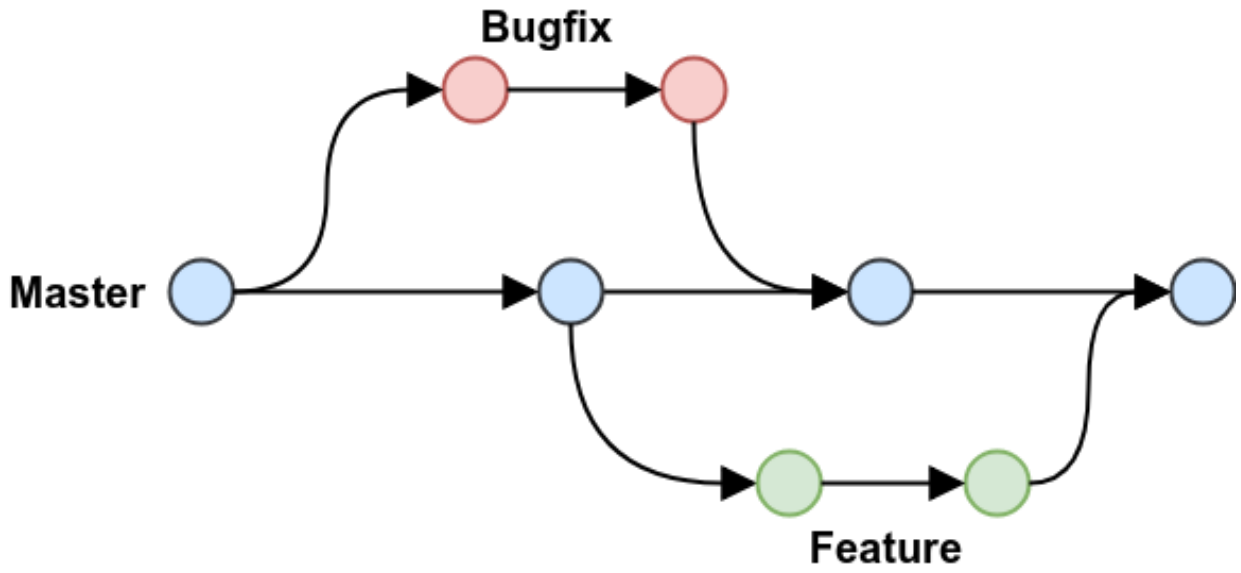

Рис 2.4 – Розділення гілок (GitHub Flow)

3 ПРОЕКТУВАННЯ МЕРЕЖЕВОГО WEB-РЕСУРСУ

3.1 Проектування мережевого WEB-ресурсу за допомогою методології функціонального моделювання SADT

При проектуванні мережевого ресурсу компанії, яка надає послуги з обліку коробок, була обрана методологія функціонального моделювання SADT (Structured Analysis and Design Technique), зокрема стандарт IDEF0. Вибір цієї методології не був випадковим; він базувався на ретельному аналізі переваг, які вона надає, а також на її здатності вирішувати складні завдання, пов'язані з розробкою інформаційних систем.

Методологія SADT, і особливо її стандарт IDEF0, забезпечує структурований підхід до системного аналізу та проектування. Це надзвичайно важливо для проектів, які мають складну ієрархічну структуру та включають багато різних функціональних компонентів. Використання IDEF0 дозволяє чітко визначити всі функції системи, розділити їх на підфункції та встановити зв'язки між ними. Для системи обліку це означає можливість детального опису кожного етапу роботи з - від надходження їх на склад до остаточної відправки замовнику. IDEF0 також дозволяє моделювати систему на різних рівнях деталізації. Це означає, що на початковому етапі можна створити загальну картину системи, а потім поступово деталізувати кожен її аспект. Такий підхід забезпечує логічність і послідовність у розробці, дозволяючи вчасно виявляти і вирішувати потенційні проблеми. В контексті системи обліку коробок це особливо важливо, оскільки дозволяє врахувати всі можливі сценарії і забезпечити її надійну і ефективну роботу.

Отже, вибір методології SADT і стандарту IDEF0 для проєктування мережевого WEB-ресурсу компанії, яка надає послуги з обліку поставок, був обґрунтованим і виваженим рішенням. Ця методологія забезпечує структурований і зрозумілий підхід до розробки, дозволяє моделювати систему на різних рівнях деталізації, сприяє ефективній комунікації між видами робіт і

оптимізує використання ресурсів. Вона дозволяє створити систему, яка буде відповідати всім вимогам, забезпечувати надійну і ефективну роботу, а також бути зрозумілою і зручною для всіх користувачів.

Контекстна діаграма мережевого WEB-ресурсу компанії, яка надає послуги у обліку, наведена на рис. 3.1.

Рисунок 3.1 – Контекстна діаграма WEB-ресурсу

Після початкового опису системи в цілому, відбувається процес функціональної декомпозиції. Цей процес розбиває кожен великий фрагмент системи на більш дрібні складові, що дозволяє отримати чітке уявлення про всі аспекти роботи системи. Завдяки цьому можна забезпечити детальний аналіз і глибоке розуміння її функціонування. Діаграми, що ілюструють розподіл і взаємодію окремих фрагментів, називаються діаграмами декомпозиції. Під час функціональної декомпозиції контекстної діаграми мережевого WEB-ресурсу

компанії, яка займається обліком коробок у складській системі, ми отримуємо три основні блоки робочих елементів системи. Кожен з цих блоків представляє собою конкретну функціональну частину, що відповідає за певний аспект роботи складу, що дає можливість забезпечити комплексний підхід до управління та оптимізації процесів зберігання та обліку товарів. Такий підхід дозволяє розглядати всі деталі системи, що підвищує ефективність та надійність роботи складської системи. Завдяки детальним діаграмам декомпозиції можна легко ідентифікувати та виправити будь-які потенційні проблеми в системі, що в кінцевому результаті сприяє поліпшенню якості обслуговування та задоволенню потреб клієнтів.

Декомпозиція кожного великого фрагмента системи на більш дрібні частини триває до тих пір, поки не досягається необхідний рівень деталізації. На цьому етапі кожна функція або процес розглядаються у контексті їх окремих компонентів та їх взаємодії з іншими частинами системи. Такий підхід дозволяє виявляти потенційні проблеми та вузькі місця на ранніх стадіях розробки. Крім того, це забезпечує чітке уявлення про те, як різні частини системи будуть взаємодіяти між собою, що сприяє кращій організації та оптимізації процесів.

Важливо, що синтаксис опису системи в цілому та кожного її фрагмента є однаковим у всій моделі. Це забезпечує узгодженість та зрозумілість усього проекту. Після проведення декомпозиції контекстної діаграми мережевого ресурсу компанії, яка займається обліком коробок у складській системі, ми отримуємо три основні блоки (рис 3.2). Кожен з цих блоків представляє собою конкретну функціональну частину системи, що відповідає за певний аспект роботи складу.

Такий підхід до моделювання дозволяє створити комплексну систему, де всі елементи працюють узгоджено і ефективно. Завдяки детальному аналізу кожного блоку ми можемо забезпечити оптимізацію всіх процесів зберігання та обліку товарів, що сприяє підвищенню ефективності роботи складу. Розуміння взаємодії між різними компонентами системи допомагає запобігти можливим конфліктам та помилкам, що, в свою чергу, підвищує надійність і стійкість всієї

системи. Це особливо важливо для складських систем, де точність і швидкість обробки інформації є критично важливими для успішного функціонування.

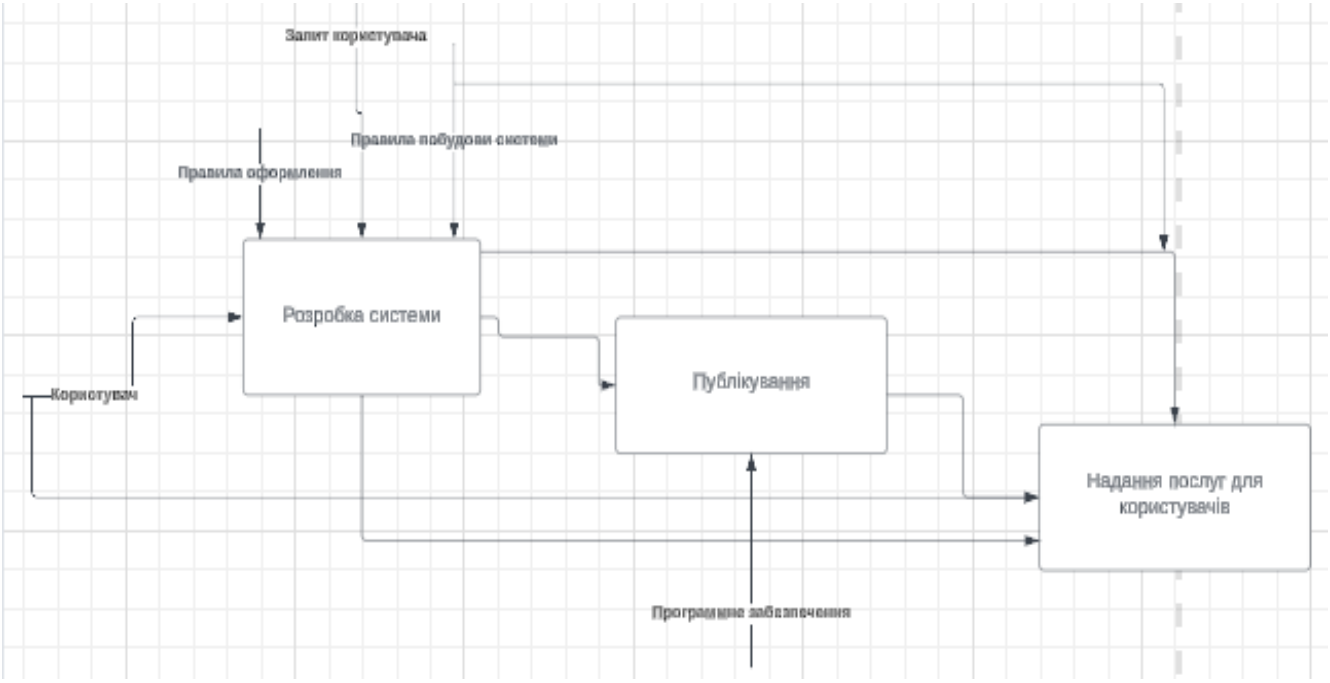

Рисунок 3.2 – Діаграма декомпозиції мережевого ресурсу

Процес "Розробка системи" передбачає створення інформаційної системи, яка працюватиме на локальному комп'ютері. Включає розробку користувацьких інтерфейсів, написання скриптів і створення баз даних, що забезпечать правильне функціонування системи. Основними вхідними даними для цього процесу є правила оформлення. Управління процесом відбувається відповідно до правил оформлення, клієнтських запитів та принципів побудови системи. Програмне забезпечення, необхідне для розробки, слугує основним інструментом, а кінцевим результатом є готова до використання інформаційна система. Це дозволяє забезпечити належне виконання всіх функцій та відповідати очікуванням користувачів.

Наступний етап, "публікування", включає розміщення системи на хостингу. Вхідними даними для цього процесу є готова інформаційна система, яку необхідно розмістити. Процес регулюється встановленими правилами, а

основним інструментом є програмне забезпечення. Завдяки цьому кроку, система стає доступною для користувачів через інтернет, забезпечуючи можливість взаємодії з нею.

Цей етап є важливим для забезпечення функціонування системи в реальних умовах, роблячи її доступною та готовою до використання на хостингу. Робота «надання послуг для користувача» призначена для того, щоб згідно з правилами оформлення, замовник міг залишитися з виконаною послугою.

Ця робота має два основні входи: дані користувача та вже доступна система, розміщена на хостингу. Керування цим процесом відбувається згідно з правилами оформлення замовлення та умовами, встановленими Інтернет-провайдером. Як інструмент використовується програмне забезпечення. Виходом цього процесу є забезпечення доступу користувачів до ресурсу, що дозволяє їм взаємодіяти з системою та використовувати її можливості. Це критично важливо для гарантування, що система працює безперебійно і відповідає всім вимогам користувачів.

На наступному етапі декомпозиції системи було виділено кілька ключових блоків першого А1 блоку. Перший блок, "розробка дизайну", використовує як вхідні дані правила оформлення, консультації з клієнтом та правила побудови системи. Основним механізмом є програмне забезпечення, а кінцевий результат - це розроблений прототип інтерфейсу для мережевого ресурсу компанії, яка займається обліком.

Далі йде блок "Розробка системи", де вхідними даними є прототип макету з попереднього етапу, а також запити від клієнта. Управлінням процесу слугують правила оформлення, вимоги клієнта і правила побудови системи. Механізмом, як і в інших етапах, виступає програмне забезпечення. Кінцевий результат цієї роботи - повністю готова інформаційна система з необхідним функціоналом.

Третій блок, "створення БД", починається з готової функціональної системи. Цей етап керується правилами побудови системи та правилами оформлення. Виконання роботи забезпечується програмним забезпеченням, а результатом є повністю функціонуюча і готова до використання база даних.

Таким чином, кожен з етапів розробки має чітко визначені вхідні дані, методи управління і механізми виконання, що дозволяє забезпечити злагоджений процес створення ефективної і надійної інформаційної системи для компанії, яка надає послуги з обліку. Завдяки такому підходу, можна виявити потенційні проблеми на ранніх стадіях і забезпечити безперебійну роботу кінцевого продукту.

Наступний етап – «тестування системи». Для цього процесу потрібні два вхідні компоненти: функціонуюча та готова система, а також інформація від клієнта. Процес тестування керується встановленими правилами оформлення та будівлі системи. Як і на попередніх етапах, основним інструментом тут виступає програмне забезпечення. Результатом цієї роботи є готова до використання система, яка пройшла всі необхідні тестування та відповідає вимогам клієнта і стандартам якості. Тестування дозволяє виявити й виправити будь-які недоліки, щоб кінцевий продукт відповідав всім очікуванням і потребам клієнта.

Діаграма декомпозиції першого А1 блоку – «розробка» представлена на рис. 3.3.

У процесі декомпозиції другого блоку А2, що стосується "публікування", було виділено чотири ключові етапи. Перший етап, "реєстрація домену", починається з отримання готової інформаційної системи та керується вимогами хостингу. Завершення цього етапу приводить до успішної реєстрації домену. На другому етапі, "реєстрація хостингу", потрібні вже зареєстрований домен та інформаційна система, керовані хостингом. Результатом цього етапу є зареєстрований хостинг.

Третій етап, "прив'язка домену", передбачає наявність зареєстрованого хостингу та інформаційної системи, яка управляється хостингом. Завершення цього етапу забезпечує прив'язку домену до системи. Нарешті, заключний етап, "публікація", включає отримання закріпленого доменного імені та інформаційної системи, яка управляється хостингом. Результатом цього етапу є ресурс, доступний у мережі Інтернет. Діаграма декомпозиції другого блоку А2, що стосується "публікування", представлена на рис. 3.4.

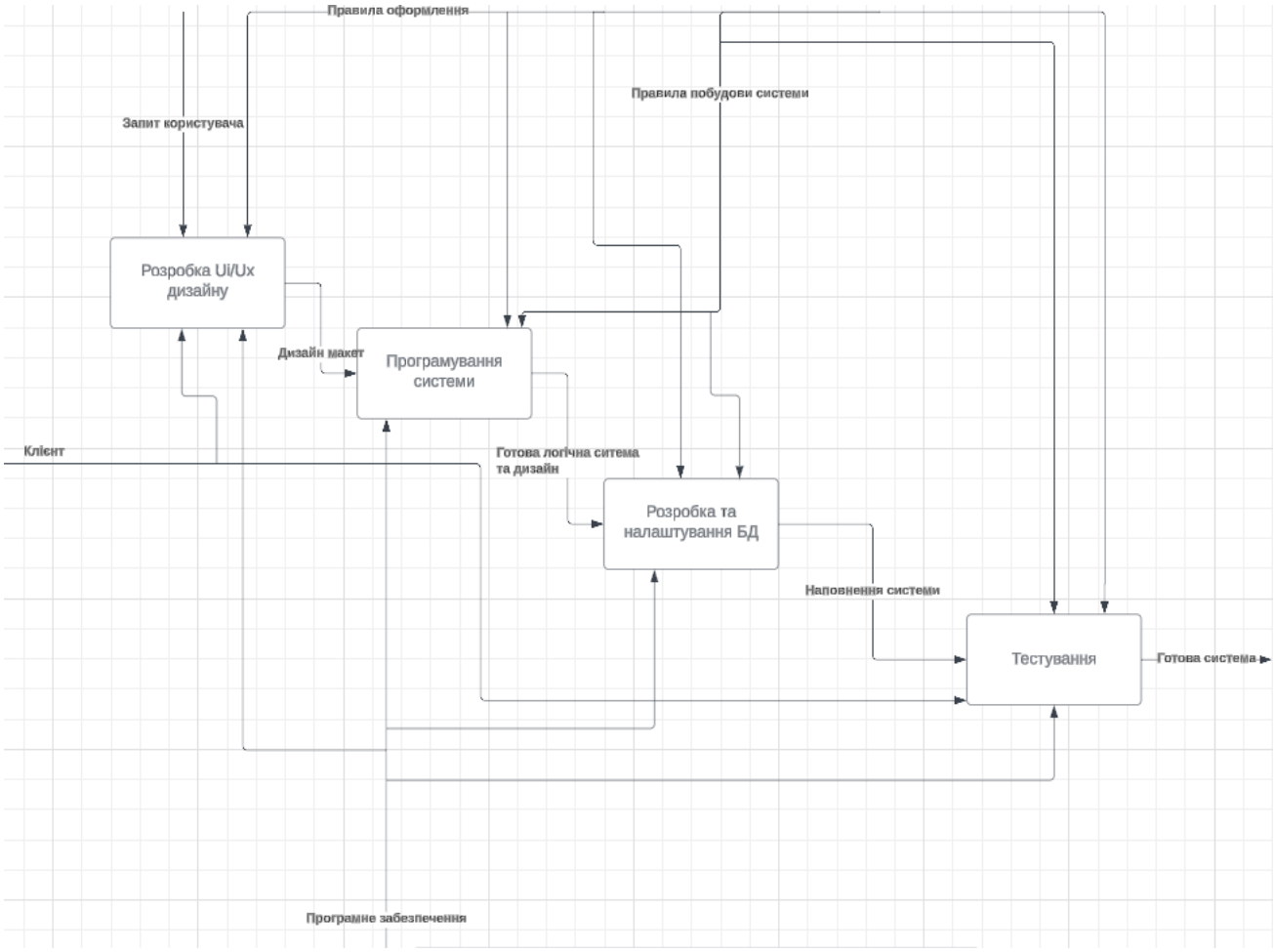

Рисунок 3.3 – Діаграми декомпозиції блоку «розробка системи»

Цей процес охоплює не лише технічні аспекти, але й забезпечує безперебійне функціонування веб-ресурсу. Під час реєстрації домену важливо враховувати всі можливі вимоги та обмеження, які можуть виникнути. На наступному етапі, реєстрація хостингу, необхідно ретельно вибрати провайдера, який зможе забезпечити надійну роботу системи. Прив'язка домену до сайту – це критичний етап, який гарантує доступність ресурсу для користувачів.

Важливо зазначити, що кожен з цих етапів має свої особливості та вимоги. Наприклад, під час реєстрації домену необхідно звертати увагу на унікальність імені та його відповідність вимогам хостинг-провайдера. Під час реєстрації хостингу, вибір надійного провайдера є ключовим для забезпечення стабільності та безпеки роботи веб-ресурсу. Прив'язка домену до сайту є важливою для

забезпечення доступу користувачів до ресурсу, а публікація веб-ресурсу дозволяє зробити його доступним в Інтернеті. Завершальним кроком є публікація, що робить веб-ресурс доступним для всіх користувачів Інтернету.

Цей процес гарантує, що веб-ресурс буде функціонувати стабільно та ефективно, забезпечуючи користувачам зручний доступ до інформації та послуг.

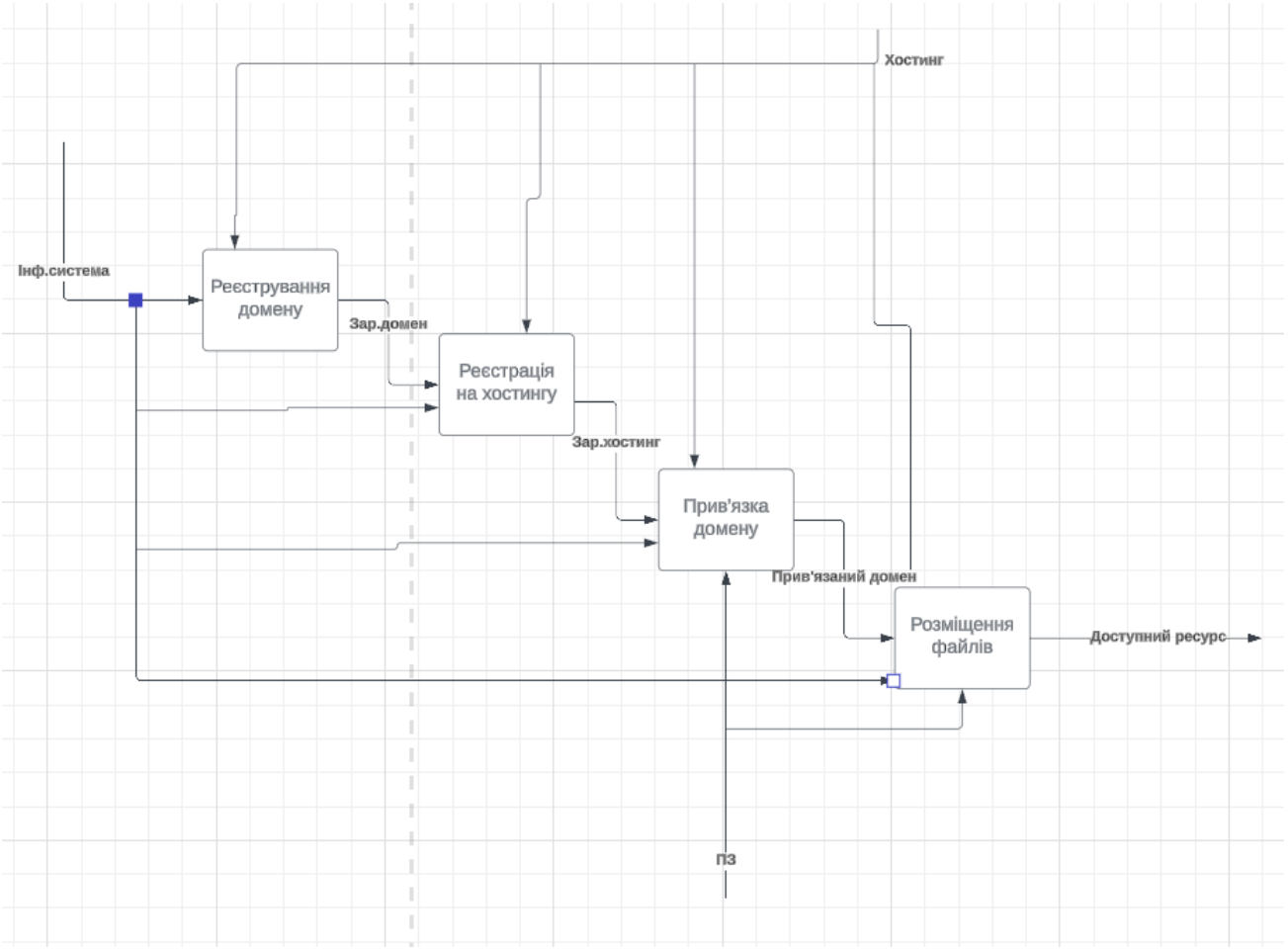

Рисунок 3.4 – Діаграми декомпозиції блоку «публікація»

В подальшому було проведено наступний етап здійснення деком позиції інформаційної системи. При декомпозиції третього А3 блоку – «надання послуг» виділені наступні три роботи.

Наступний етап декомпозиції інформаційної системи полягав у розбитті третього блоку А3 – «надання послуг» – на окремі частини. В результаті цього процесу було визначено три основні завдання.

Першим завданням є "реєстрація користувачів". Для цього потрібні дані користувачів та доступ до веб-ресурсу в мережі Інтернет. Процес керується правилами оформлення і призводить до успішної реєстрації користувача в системі.

Друге завдання – "доступ до функцій". Зареєстровані користувачі мають можливість користуватися функціями системи, такими як додавання поставок, їх редагування та відстеження.

Третє завдання – "оформлення". Цей процес включає організацію обліку, керується правилами оформлення, і його результатом є завершена послуга для клієнта.

Ця деталізація допомагає зрозуміти кожний етап роботи інформаційної системи відокремлено. Реєстрація користувачів становить основу для подальшого доступу до системи, що дозволяє їм повноцінно взаємодіяти з нею. Надаючи користувачам доступ до функцій системи, ми забезпечуємо їм можливість ефективного керування своїми поставками, що є важливим для зручного користування системою. Останній етап, оформлення обліку, гарантує надання послуг користувачам у відповідності з встановленими правилами, що допомагає забезпечити якісне виконання замовлень та задоволення їх потреб.

Кожен з цих етапів важливий для забезпечення повноцінного функціонування системи, де кожен процес чітко структурований і взаємопов'язаний, що сприяє досягненню високої ефективності та надійності.

Діаграма декомпозиції третього А3 блоку – «надання послуг» наведена на рис. 3.5.

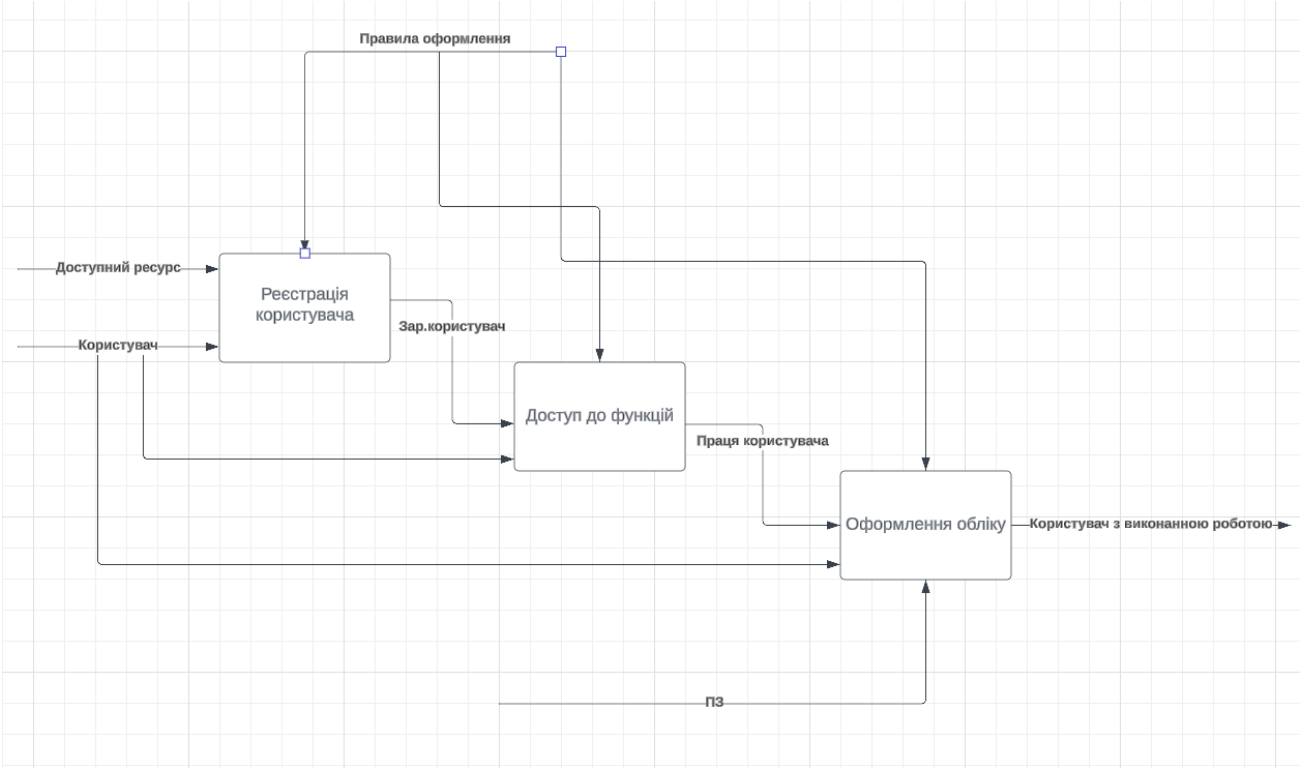

Рисунок 3.5 – Діаграми декомпозиції блоку «надання послуг»

3.2 Проектування мережевого WEB-ресурсу за допомогою методології Workflow Diagramming

Під час подальшого проектування мережевого веб-ресурсу компанії, яка спеціалізується на обліку поставок, було використано методологію послідовного виконання процесів, відому як Workflow Diagramming або стандарт IDEF3. Ця методологія дозволяє нам зрозуміти логіку виконання різних дій у системі. IDEF3 можна використовувати як самостійно, так і в поєднанні з іншими методологіями, такими як IDEF0.

Кожен функціональний блок в IDEF0 може бути подано у вигляді послідовності процесів або операцій за допомогою IDEF3.

Якщо в IDEF0 вказано, що система робить, то IDEF3 допомагає зрозуміти, як саме це відбувається.

За методологією IDEF3, головна функція мережевого веб-ресурсу компанії, що спеціалізується на обліку поставок, полягає у "наданні послуг та підтримці у обліку поставок". Цю основну функцію можна відобразити на контекстній діаграмі системи. Після проведення декомпозиції цієї діаграми можна спостерігати за послідовністю виконання різних завдань, яка зображена на рисунку 3.6.

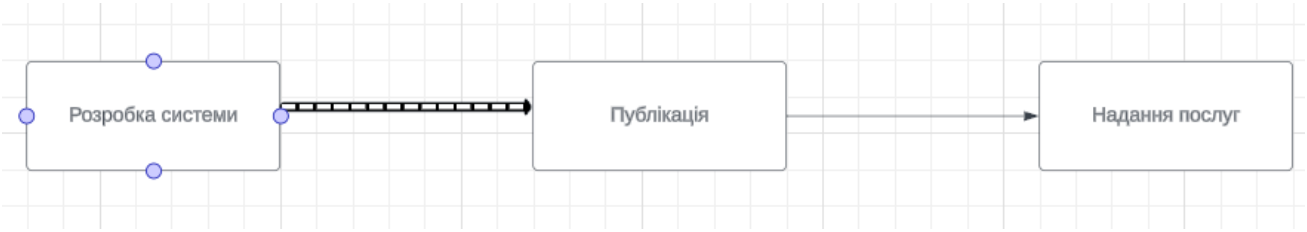

Рисунок 3.6 – Діаграма декомпозиції мережевого ресурсу компанії

Спочатку розпочинається процес "розробки системи", який передує виконанню роботи "публікація". Послідовність цих робіт показує, що робота-приймач може завершитися не лише після завершення роботи-джерела. Наступним етапом є робота "надання послуг", яка має старший зв'язок із блоком "публікація", передбачаючи завершення всіх робіт, що виконуються перед нею. Під час подальшої декомпозиції роботи "розробка системи" ми отримуємо чотири блоки, включаючи роботи з двома перехрестями(рис. 3.7).

Після закінчення першої роботи «розробка UI/UX інтерфейсу» наступає перше перехрестя «асинхронне І». Це означає, що наступні завдання можуть розпочатися не одночасно, але вони повинні бути запущені. Ці завдання включають «розробку системи» та «розробку бази даних». Під час об'єднання стрілок-виходів з цих завдань використовується те ж саме перехрестя «асинхронне І», що показує, що завдання можуть завершитися, але це може статися не одночасно. Далі виконується робота «тестування системи», яка є завершальною.

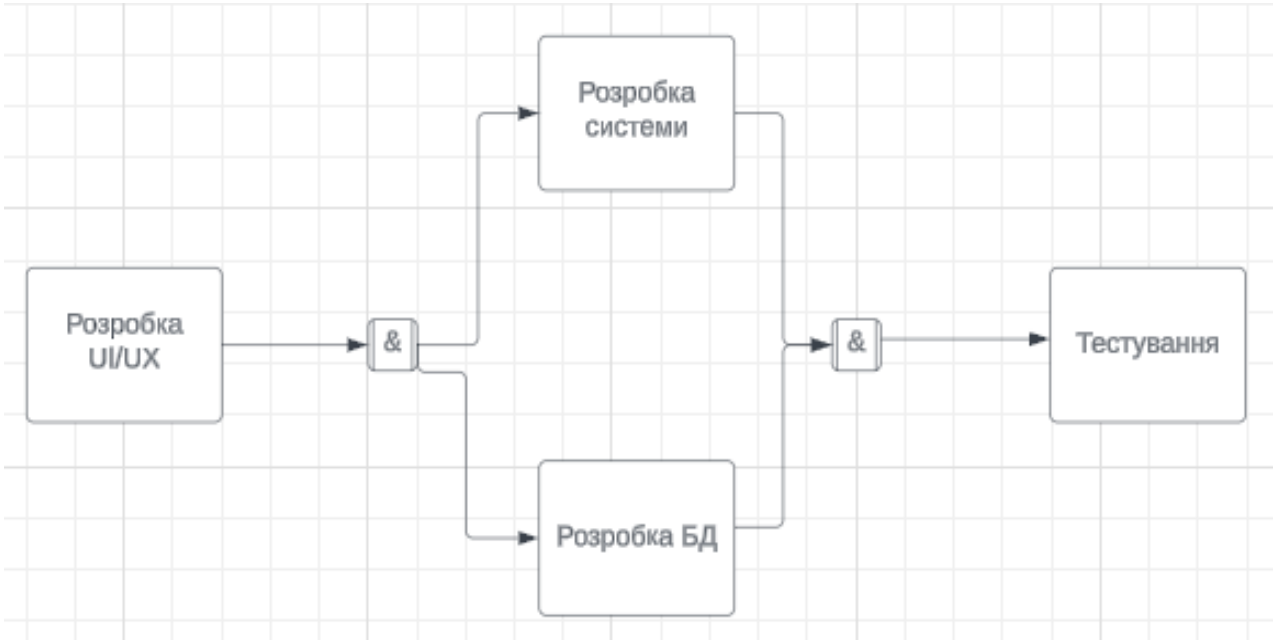

Рисунок 3.7 – Діаграма декомпозиції блоку «розробка WEB-системи»

Рисунок 3.8 – Діаграма декомпозиції роботи «публікування»

Під час аналізу «видавничого» процесу встановлено, що всі етапи забезпечені перевагами та мають старше відношення між собою. Перший етап — «реєстрація домену», потім «реєстрація хостингу», потім «прив’язка домену до сайту» і, нарешті, етап «файли хостингу». (рис. 3.8).

При аналізі роботи над наданням послуг виявлено, що її можна розділити на три блоки робіт, які виконуються послідовно та мають між собою старший зв'язок. Кожен з цих блоків виконується у певній послідовності, що забезпечує ефективність і надійність процесу. (рис. 3.9).

Рисунок 3.9 – Діаграма декомпозиції роботи «надання послуг»

3.3 Проектування мережевого WEB-ресурсу за допомогою методології потоків даних DFD

У контекстній діаграмі основним процесом системи є "надання інформації". Зовнішні сутності, такі як "користувач" і "сервер", впливають на систему. Також є блок-сховище даних. Зв'язок між замовником і головною роботою відбувається через "запит". Дані з системи надходять до зовнішніх сутностей, а від "сервера" до "інформаційної системи". Облікові дані передаються до системи зі сховища даних. Контекстна діаграма мережевого веб-ресурсу компанії, яка спеціалізується на наданні послуг у обліку, представлена на рисунку 3.10.

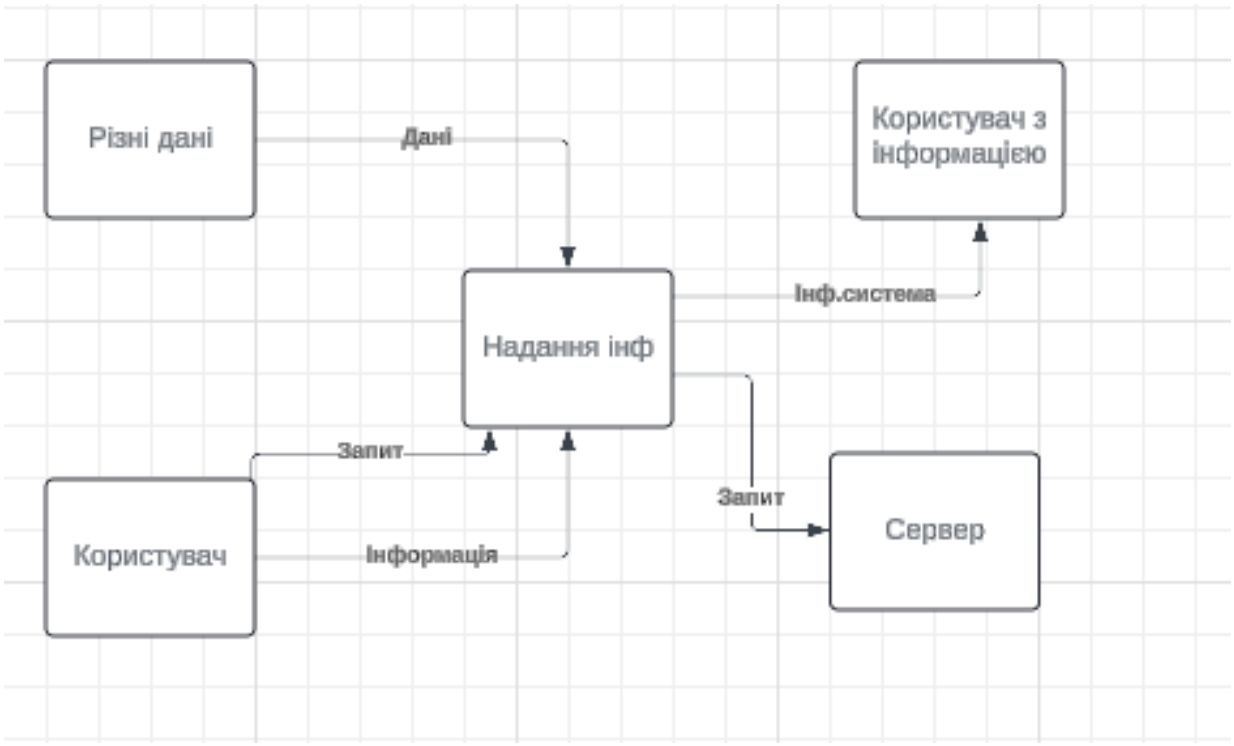

Рисунок 3.10 – Контекстна діаграма

Головний процес, зображений на контекстній діаграмі, розглядається через декомпозицію. На першому рівні ієрархії показані основні внутрішні процеси системи та їх зв'язки з зовнішніми сутностями. Перший процес – "розробка системи", приймає дані від зовнішньої сутності "користувач" та зі сховища даних. Вихідні дані цього процесу – "інформаційна система".

Другий блок – "хостинг" – отримує дані від "розробки системи" та "сервера", виводить "файли" та створює "WEB-ресурс", який стає входом для зовнішньої сутності "купівля".

Останній блок – "купівля". Дані від "користувача" включають "дані про коробки та поставки" та "запит", від "хостингу" – "WEB-ресурс", від "сервера" – "файли", та з блоку "дані" – "облікові дані". Результатом є "дані", що надсилаються зовнішній сутності "користувач з наданою інформацією". Діаграма декомпозиції подана на рисунку 3.11.

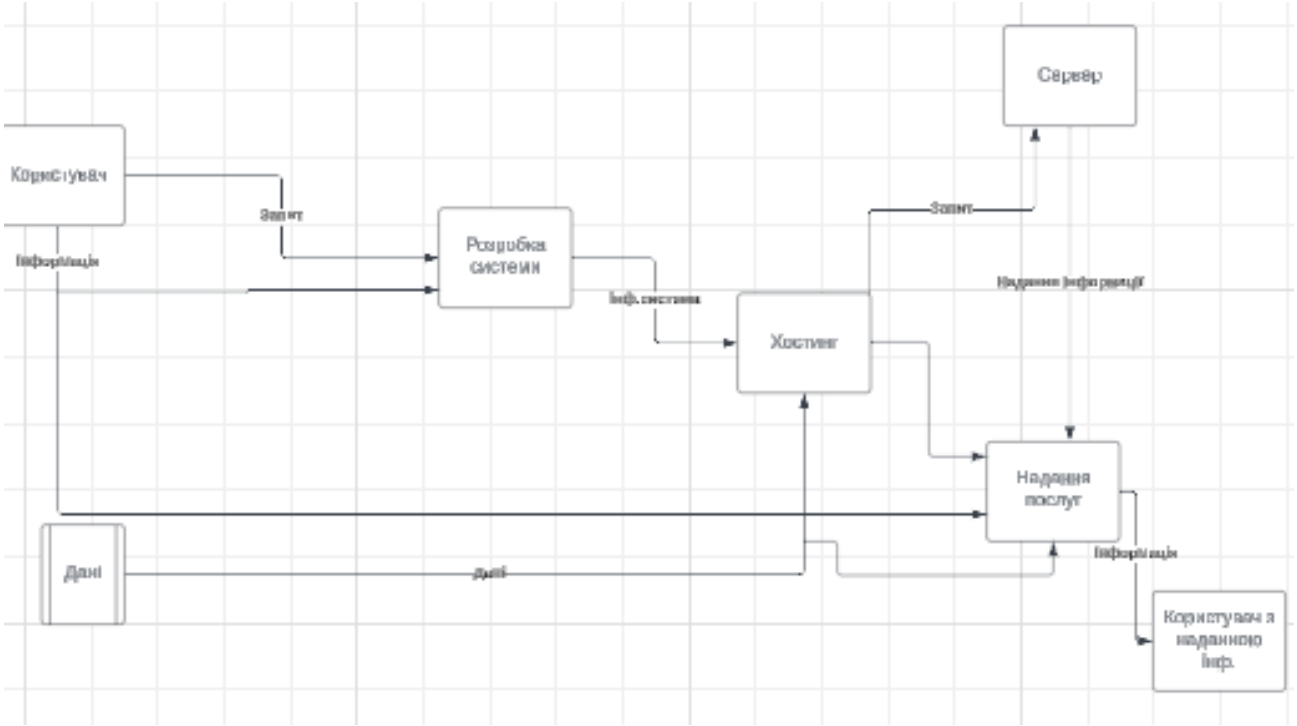

Рисунок 3.11 – Діаграма декомпозиції ІС для надання послуги

Завдяки застосуванню методології потоків даних в проектуванні інформаційної системи для обліку послуг, опис процесу обробки інформації в

системі став можливим. Ця система спрямована на забезпечення ефективного виконання облікових функцій. Її розробка виконувалася як додаток до методології функціонального моделювання IDEF0, що дозволило узгоджувати функціональні аспекти системи з обробкою потоків даних

3.4 Проектування бази даних системи

Трирівнева архітектура "клієнт-сервер" була обрана для мережевого WEB-ресурсу компанії з фотозйомки та пост обробки з декількох причин. По-перше, ця архітектура дозволяє ефективно розділити функціональні обов'язки між клієнтською і серверною частинами системи. Клієнтська програма-браузер на робочих станціях дозволяє користувачам легко інтерактивно взаємодіяти з системою через веб-інтерфейс.

По-друге, розміщення бази даних на сервері локальної комп'ютерної мережі забезпечує централізований доступ до даних для всіх користувачів системи. Це дозволяє забезпечити їхню консистентність та надійність, а також спрощує процес адміністрування бази даних.

Крім того, така архітектура дозволяє ефективно масштабувати систему в майбутньому, в разі потреби у збільшенні обсягів обробки даних або кількості користувачів. Вона також забезпечує більшу безпеку, оскільки доступ до бази даних може бути контрольований і обмежений на рівні сервера.

Перший крок у створенні бази даних - це аналіз вимог і потреб користувачів. Це важливий етап, оскільки від нього залежить подальший дизайн і функціональність бази даних. Під час аналізу, потрібно зрозуміти, які дані потрібно зберігати, які зв'язки між ними існують, і як користувачі будуть взаємодіяти з системою.

Після аналізу вимог відбувається проектування бази даних. На цьому етапі визначається структура даних, таблиці, поля та зв'язки між ними. Важливо враховувати нормалізацію даних для забезпечення їх консистентності та ефективного використання.

Після того, як структура бази даних визначена, настає етап реалізації. Це означає створення самої бази даних за допомогою відповідної СУБД та мови запитів. На цьому етапі створюються таблиці, індекси, зберігаються процедури та функції, які будуть використовуватися для роботи з даними.

Останнім етапом є тестування і оптимізація бази даних. Під час тестування перевіряється правильність роботи всіх функцій та операцій бази даних, а також її продуктивність. Якщо виявляються помилки або можливості для поліпшення, вони виправляються та вдосконалюються в процесі оптимізації.

Сутність ПОСТАВКА містить наступні атрибути (табл. 1): id, а також вона містить кількість коробок.

Таблиця 1 – Сутність «Поставка»

№	Атрибут	Тип
1	id	var(32)
2	howmucbpxes	Int(24)

Сутність коробка містить атрибути (табл. 2): condition, що означає стан коробки, а також до якої поставки належить, а також box_id, createdby

Таблиця 2 – Сутність «КОРОБКА»

№	Атрибут	Тип
1	condition	var(15)
2	ware_id	int(32)
3	box_id	Int(32)
4	createdby	Varchar(32)

Сутність користувач містить атрибути (табл 3): login, password,а також ID

Таблиця 3 – Сутність «Користувач»

№	Атрибут	Тип
1	login	Varchar(255)
2	password	varchar(255)
3	id	int(255)

Після створення таблиць та проведення описаного аналізу і оптимізації бази даних ми можемо зробити декілька висновків. По-перше, створення таблиць дозволило нам визначити необхідні дані та їх структуру для зберігання. Далі, процес нормалізації допоміг нам усунути надмірність та уникнути аномалій, що можуть виникнути при зберіганні даних. Оптимізація бази даних дозволила підвищити швидкість та ефективність системи, що є критичним для забезпечення продуктивної роботи та задоволення потреб користувачів. В цілому, цей процес допоміг покращити якість та надійність бази даних, що в свою чергу сприяє покращенню роботи всієї інформаційної системи.

Після завершення процесу створення, нормалізації та оптимізації бази даних можна впевнено сказати, що база даних готова до використання в реальних умовах. Однак, важливо пам'ятати, що робота з базою даних не закінчується на етапі її створення. Постійний моніторинг та підтримка бази даних є необхідними для забезпечення її стабільної роботи.

У процесі експлуатації можуть виникати різні ситуації, що потребують додаткових змін або коригувань. Наприклад, зміни в бізнес-процесах можуть вимагати додавання нових таблиць або полів, зміни у зв'язках між даними.

Завдяки трирівневій архітектурі «клієнт-сервер», наша система має додаткові переваги у вигляді розподілу навантаження між клієнтськими комп'ютерами і сервером. Це підвищує загальну ефективність роботи системи і забезпечує швидкий відгук на запити користувачів. Крім того, така архітектура дозволяє легше впроваджувати нові функціональні можливості та здійснювати

технічну підтримку системи, що є важливим для довготривалої стабільної роботи і розвитку компанії.

В результаті створення, нормалізації та оптимізації бази даних, ми отримали добре структуровану, ефективну та надійну систему для зберігання і обробки даних. Цей процес є критично важливим для забезпечення стабільної роботи інформаційної системи компанії, яка надає послуги у сфері обліку поставок. Оптимізація бази даних дозволила досягти високої продуктивності та швидкості обробки запитів, що є ключовим для забезпечення задоволеності користувачів та успішного функціонування бізнесу.

Безпека бази даних та її регулярне резервне копіювання стали основою для запобігання втратам даних та захисту від несанкціонованого доступу. Постійний моніторинг та підтримка бази даних гарантують її стабільну роботу, а також дозволяють швидко реагувати на зміни в бізнес-процесах та забезпечувати актуальність даних.

Таким чином, належним чином спроектована та оптимізована база даних є невід'ємною частиною успішної інформаційної системи, що дозволяє компанії ефективно виконувати свої завдання, досягати бізнес-цілей та задовольняти потреби своїх клієнтів.

4 ПРАКТИЧНА РЕАЛІЗАЦІЯ МЕРЕЖЕВОГО WEB-РЕСУРСУ

4.1 Керівництво додатком користувача-клієнта системи

Розроблений мережевий ресурс для компанії, яка надає послуги у сфері обліку поставок, є важливим етапом у процесі створення сучасної інформаційної системи, що сприяє автоматизації та оптимізації бізнес-процесів. Така система не тільки забезпечує присутність компанії в Інтернеті, але й створює інтерактивну платформу, яка дозволяє клієнтам легко взаємодіяти з компанією, здійснювати управління своїми поставками та отримувати необхідні звіти в режимі реального часу. Крім того, інтерактивні можливості WEB-ресурсу сприяють залученню нових клієнтів, покращенню якості обслуговування та підвищенню рівня задоволеності існуючих клієнтів. Завдяки зручному і інтуїтивно зрозумілому інтерфейсу користувачі можуть легко здійснювати необхідні операції, що сприяє зростанню довіри до компанії і зміцненню її позицій на ринку.

Мережевий WEB-ресурс для системи обліку поставок відіграє ключову роль для користувачів, надаючи їм зручний та ефективний інструмент для управління поставками. Користувачі можуть реєструватися на платформі, отримувати доступ до всіх функцій системи.

Завдяки інтерактивному інтерфейсу, користувачі мають можливість оперативно отримувати необхідну інформацію, що дозволяє їм приймати обґрунтовані рішення у реальному часі.

Ця система не тільки спрощує процес обліку поставок, але й забезпечує зручний доступ до актуальних даних, що сприяє підвищенню ефективності роботи компанії.

У цьому розділі буде докладно розглянуто створений ресурс. Головна сторінка системи представлена на рис. 4.1.

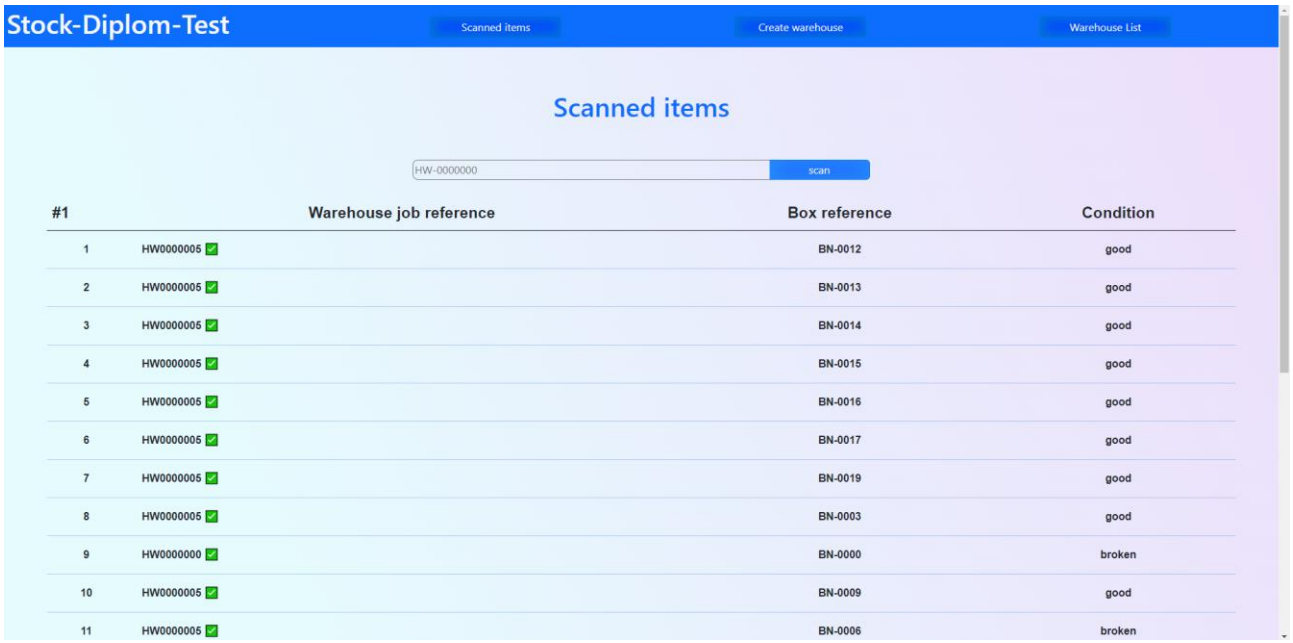

Рисунок 4.1 – Головна сторінка ІС

У верхній частині системи розміщене меню, на якому знаходиться посилання на пункти головного меню: перша кнопка - це актуально (головна) сторінка, друга – це посилання на сторінку створення нової поставки, а третя – це більш детальний список поставок з трохи іншими можливостями. Зокрема на цій сторінці є зелена кнопка редагування. Ця кнопка відповідає за редагування кожної окремої коробки. Наприклад, якщо вже після поставки було визначено, що товар ушкоджений – коробку можна перепомітити як ушкоджену. У строках цієї таблиці присутня наступна інформація: номер поставки, який є абсолютно унікальним, а також номер коробки, який є також унікальним. Важливо відмітити, що кожен номер коробки прив’язаний до номеру поставки, тому данні не розташованні у випадковому форматі, все структуровано та зрозуміло для кінцевого користувача. Зверху є поле «скан», але наразі воно виконує функцію оптичного сканеру. Зокрема треба зазначити, що кожна поставка не може містити в собі 20 коробок. Ця функція створена для того щоб не перенасичувати інтерфейс і не плутати користувача, тому що цей ресурс створений переважно для малих бізнесів.

Як вже було зазначено, на сторінку редагування коробки, можливо перейти лише за допомогою спеціальної зеленої кнопки. Сторінка редагування представлена на рисунку 4.2.

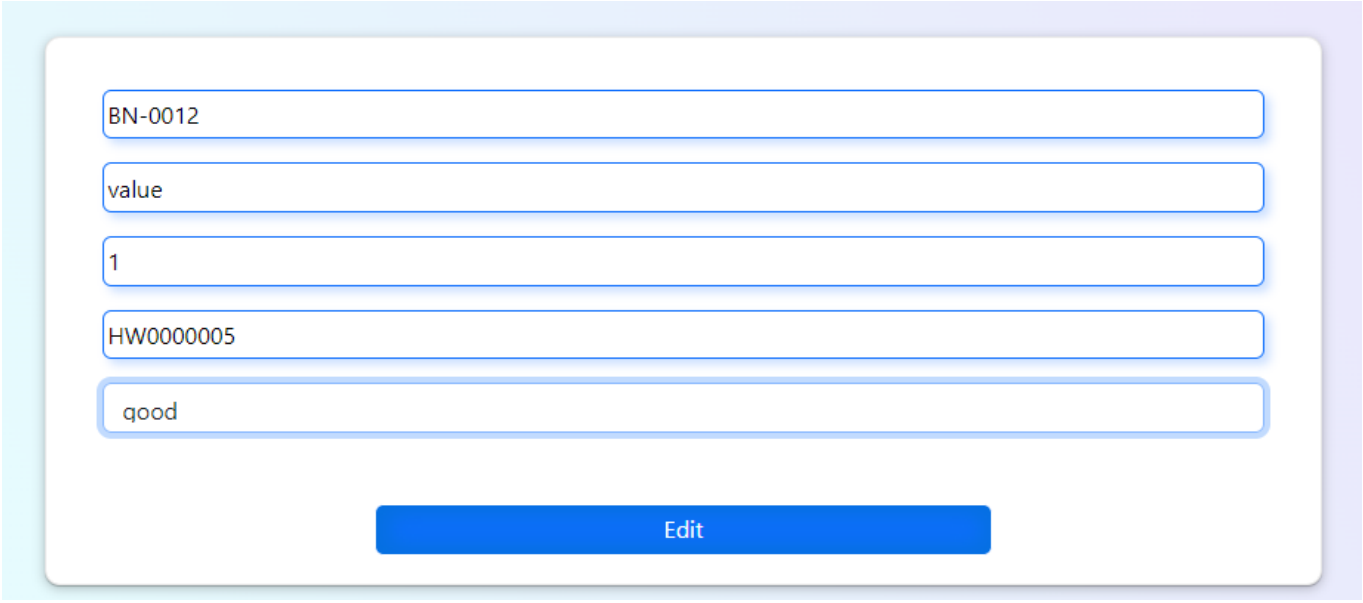

Рисунок 4.2 – Сторінка редагування

На цій сторінці представлено 5 полів для взаємодії. Перще поле – це унікальний ідентифікатор, друге – це опис, третє – кількість предметів у коробці, четверте – це унікальний номер поставки до якого прив’язана коробка, а останнє п’яте поле – це стан товару у коробці. Нижче представлена кнопка, що дозволяє зберегти зміни. Всі ці поля є інтерактивними, а останнє поле – це спеціальне меню яке надає змогу змінити стан тільки на два варіанти, а саме на ушкоджений стан, та цілком цілий (рис 4.3).

Наступна сторінка – це сторінка створення поставки. На цій сторінці було створено 4 незалежних поля, а саме:

— Унікальний ідентифікатор: цей індентифікатор має в собі 17 символів з яких 13 є унікальними. Цей ідентифікатор можливо створити вручну, а можна залишити цей нелегкий труд на систему, яка згенерую його

автоматично. Якщо ідентифікатор буде невірним з якихось причин – нова поставка просто не створиться;

— Поле номеру поставки: цей номер э також абсолютно унікальним і повинен мати в собі 9 символів, з яких 7 повинні бути унікальними;

— Локація: це поле визначає звідки та від кого прийшла поставка. Це допоже користувачу ефективно відслідковувати поставку, а також знаходити контакти, якщо з поставкою сталося щось не так. Для цього поля немає валідації, тому тут можливо залишити інформацію не тільки про те звідки ця поставка, а й номер відповідального лиця;

— Поле, що визначає ким створена ця поставка. Нажаль, бувають такі випадки, коли потрібно визначити хто був відповідальним лицем при отриманні поставки, тому тут є це поле.

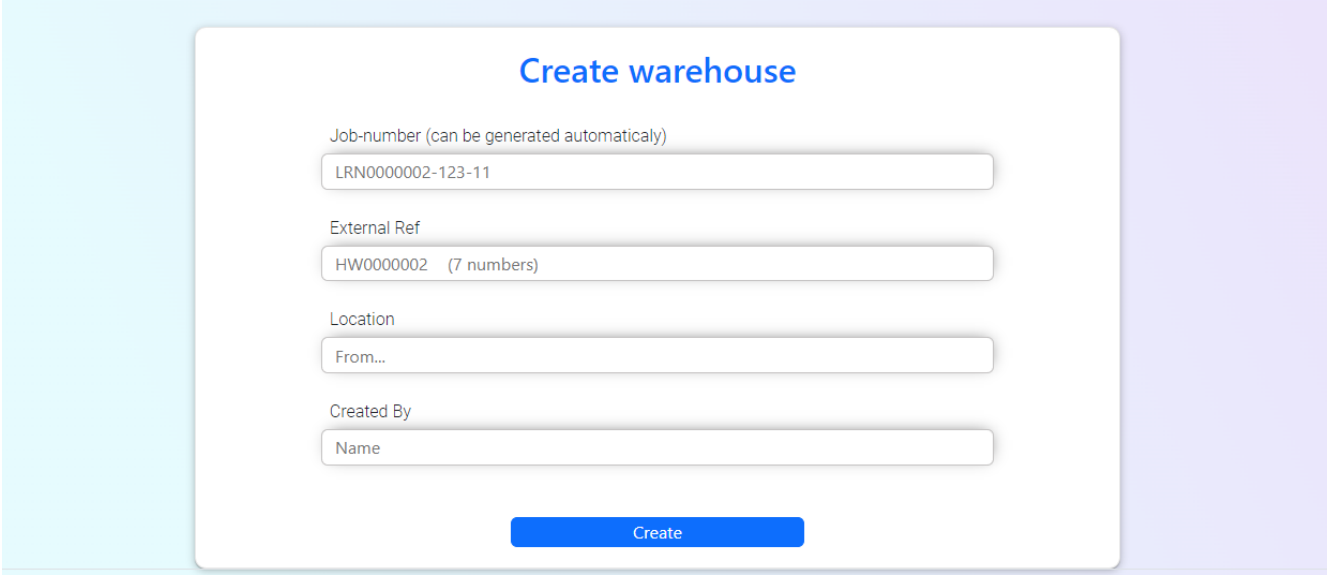

Рисунок 4.3 – Сторінка створення поставки

Після натиснення кнопки «створити» дані відправляються по таблицям.

Наступна сторінка – це більш детальна таблиця саме для поставок(рис 4.4). на цій сторінці можна переглянути поставки і тільки їх. Всього тут є 5 елементів таблиці, а саме:

— Елемент таблиці з унікальним ідентифікатором;

— Елемент таблиці з номером поставки;

— Елемент таблиці з локацією;

— Елемент, де вказується хто саме створив поставку.

— Елемент з інформацією з тим звідки прийшла поставка, а також інші контактні дані, які користувач може вказати у довільному порядку.

Рисунок 4.4 – Сторінка поставок

У самій правій частині таблиці є кнопка, що переводить користувача на сторінку мануального створення коробки для конкретної поставки(рис 4.5). Це потрібно для користувачів, які хочуть мати повний контроль над створенням поставки.

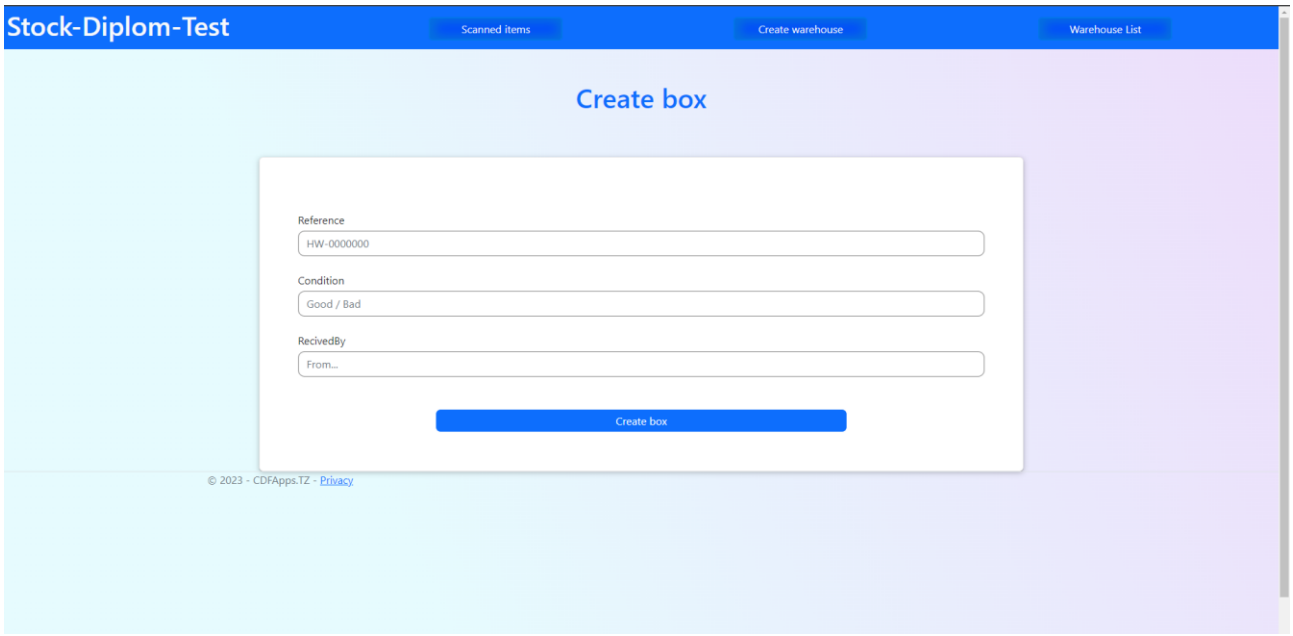

Рисунок 4.5 – Сторінка створення коробки

На сторінці створення коробки є всього 3 поля та 1 кнопка. Перше поле – це номер коробки, друге – її стан, а третє – звідки вона. Останнє значення буде передано у таблицю БД, де поставка.

Наступна сторінка сайту – це сторінка політики конфінденційності[11] (рисунок 4.5). Ця сторінка відрізняється від інших своїм унікальним дизайном. Темний зоряний фон на якому розміщений контейнер з ефектом матового скла в якому є інформація що до політики. Цей контейнер має свій окремий скролбар, тому це виглядає не тільки гарно з боку стилю, але й є зручним.

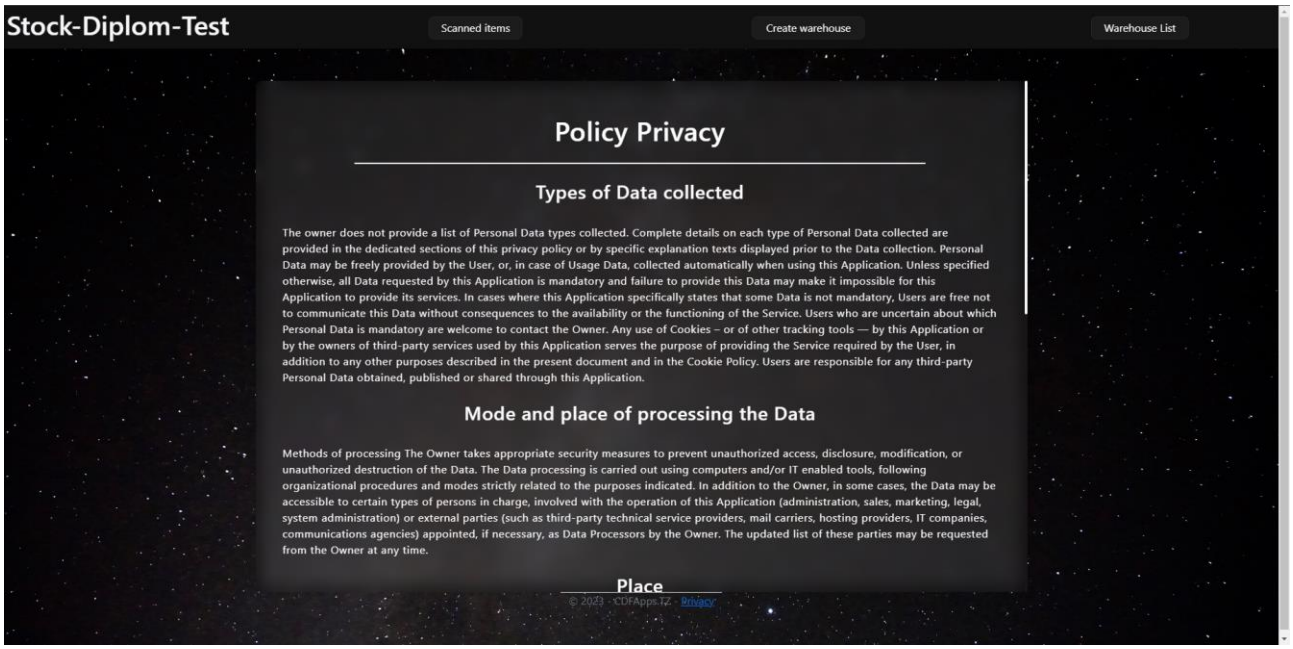

Рисунок 4.6 – Сторінка з політикою конфіденційності.

Сторінка з політикою конфіденційності на була створена для забезпечення прозорості та довіри між компанією та її користувачами. Зрозуміло, що у сучасному цифровому світі конфіденційність даних є надзвичайно важливою для кожного користувача, тому компанія прагне забезпечити максимальний захист особистої інформації користувачів. На цій сторінці детально описано, які дані можуть бути зібрані, як їх можуть використати та зберегти, а також які заходи безпеки застосовуються для їх захисту.

ВИСНОВКИ

В результаті виконання дипломної роботи було здійснено проектування та програмну реалізацію системи складу, яка забезпечує ефективне управління складськими операціями та облік товарів. Розробка такої системи дозволяє оптимізувати процеси зберігання і переміщення товарів, що значно підвищує ефективність роботи складу і зменшує витрати, пов'язані з управлінням складом. Це, в свою чергу, сприяє підвищенню задоволеності клієнтів завдяки більш швидкому і точному виконанню замовлень.

В ході дипломного проектування було проведено аналіз існуючих аналогічних систем управління складом, що дозволило виявити їх переваги і недоліки. Це дало змогу чітко визначити вимоги до розробки системи, враховуючи сучасні потреби підприємств у сфері складського обліку. Було визначено архітектуру системи, здійснено вибір і обґрунтування програмних засобів реалізації, проведено детальне проектування, а також розроблено базу даних для зберігання всіх необхідних даних про товари.

Система складу реалізована з використанням сучасних програмних засобів, що забезпечує зручний інтерфейс користувача і інтуїтивно зрозумілі діалогові вікна. Це дозволяє користувачам легко орієнтуватися в системі і ефективно виконувати свої завдання.

Використання таких технологій, як фреймоворк asp.net у парі з патерном MVC , а також використання таких мов я C#, мова сценаріїв JS, технології HTML та CSS, забезпечує надійність і масштабованість системи.

Система складу не тільки полегшує процеси управління складом, але й забезпечує можливість інтеграції з іншими системами підприємства, такими як системи обліку та управління замовленнями. Це дозволяє створити єдину інформаційну платформу, що значно підвищує оперативність і точність бізнес-процесів.

Використання такої системи складу дозволяє підприємству орієнтуватися на залучення потенційних клієнтів, покращення якості обслуговування і підвищення прибутковості. Система забезпечує повний контроль над всіма складськими операціями, що сприяє зменшенню витрат і підвищенню ефективності роботи підприємства в цілому.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Історія розвинення складських систем у Україні та світі URL: https://wareteka.com.ua/blog/kak-razvivalas-skladskaya-logistika/
2. Компанія KeyCrm URL: https://ua.keycrm.app/
3. DTNrade компанія URL: https://dntrade.com.ua/
4. Cleverence компанія URL: https://cleverence.in.ua/
5. Dilovod компанія URL: ttps://dilovod.ua
6. Коротка історія JavaScript URL: https://habr.com/ru/companies/livetyping/articles/324196/
7. Що таке фреймворк і навіщо він потрібен URL: https://brainlab.com.ua/blog/chto-takoe-frejmvork-obyasnyaem-prostymi-slovami
8. Що таке верстка та їх різновиди URL: https://webtune.com.ua/statti/web-rozrobka/verstka-sajtiv/
9. ASP. NET URL: https://learn.microsoft.com/ru-ru/aspnet/overview
10. MVC архітектура URL: https://towardsdatascience.com/everything-you-need-to-know-about-mvc-architecture-3c827930b4c1
11. Що таке політика конфіденційності та як її створювати URL: https://marketing.link/uk/yak-stvorty-polityku-konfidenczijnosti-dlya-vashogo-sajtu