

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВАДИМА ГЕТЬМАНА
Навчально-науковий інститут
«Інститут інформаційних технологій в економіці»
Кафедра інформаційних систем в економіці**

ОСВІТНЬО ПРОФЕСІЙНА ПРОГРАМА «КОМП'ЮТЕРНІ НАУКИ»
галузь знань 12 «Інформаційні технології»
спеціальність 122 «Комп'ютерні науки»

Форма навчання: денна

КВАЛІФІКАЦІЙНИЙ БАКАЛАВРСЬКИЙ ПРОЕКТ

на тему

**ПРОЄКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ ОБЛІКУ
БУДІВЕЛЬНИХ ІНСТРУМЕНТІВ**

здобувача Гончарука Владислава Сергійовича _____

Науковий керівник:

д.т.н., професор

_____Артемчук В.О.

**Проект допущений до захисту
перед екзаменаційною комісією з
атестації здобувачів вищої освіти
завідувач кафедри:**

к.е.н., доцент.

_____Тішков Б.О.

Київ 2024

Міністерство освіти і науки України
Київський національний економічний університет імені Вадима Гетьмана
Навчально-науковий інститут «Інститут інформаційних технологій в економіці»

Кафедра інформаційних систем в економіці

ОСВІТНЬО_ПРОФЕСІЙНА ПРОГРАМА «КОМП'ЮТЕРНІ НАУКИ»

галузь знань 12 «Інформаційні технології»

спеціальність 122 «Комп'ютерні науки»

ПОГОДЖЕНО:

ЗАТВЕРДЖУЮ:

Керівник проектної групи(гарант)

Завідувач кафедри

освітньо-професійної програми

_____Тішков Б.О.

_____Іванченко Г.Ф.

“ _____ ” _____ 2024 р.

“ _____ ” _____ 2024 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

здобувача вищої освіти ***Гончарука Владислава Сергійовича***

очної (денної) форми навчання

на підготовку кваліфікаційного бакалаврського проекту
на тему: «Проектування інформаційної системи обліку будівельних інструментів»

Тему затверджено наказом ректора Університету від 11.03.2024 р. № 529-ст.

Кваліфікаційний бакалаврський проект виконується на матеріалах
інтернет-ресурсів та власних розробок.

План кваліфікаційного бакалаврського проекту

Розділ I. Характеристика та аналіз предметної галузі

Розділ II. Розробка вимог і моделювання інформаційної системи

Розділ III. Проектування та реалізація компонентів системи/підсистеми

Об'єкт дослідження - інформаційна система обліку будівельних інструментів

Предмет дослідження - процеси та механізми управління і контролю за використанням будівельних інструментів

Мета кваліфікаційного бакалаврського проекту - провести дослідження і виконати проектні рішення зі створення інформаційної системи обліку будівельних інструментів та показати технологію функціонування системи

Конкретні завдання, які студент повинен виконати для досягнення поставленої мети:

У розділі I потрібно охарактеризувати предметну галузь, яка включає управління та контроль за використанням будівельних інструментів, описуючи головну мету галузі та особливості, які впливають на організацію інформаційної системи. Далі слід дати характеристику об'єкта дослідження, яким є інформаційна система обліку будівельних інструментів, описуючи, як система спрямована на ефективне управління та контроль за інструментами. Також потрібно проаналізувати існуючі задачі та перелічити програмні засоби, які використовуються для обліку будівельних інструментів. Крім того, важливо висвітлити організаційну структуру підприємства, що використовує систему, описуючи основні функції підрозділів та їх взаємозв'язки.

У розділі II потрібно розробити вимоги та здійснити моделювання інформаційної системи обліку будівельних інструментів. Спершу необхідно провести аналіз і специфікацію вимог до системи, що включає визначення функціональних та нефункціональних вимог, які забезпечать належну роботу системи. Потім слід описати постановку задачі, зокрема характеристики задачі, вхідну та вихідну інформацію. Наступним кроком є моделювання системи, що передбачає створення моделі поведінки системи, а також розподіл вимог за компонентами системи.

У розділі III виконати проектування таких компонентів системи: інформаційного, технічного та програмного забезпечення. Конкретизація завдань викладена достатньо повно в методичних рекомендаціях кафедри що до виконання бакалаврського дипломного проекту.

В цьому розділі студент має також показати результати виконання програмних модулів, вказав можливість та доцільність впровадження їх на практиці, також вказати особистий вклад. **Обов'язково** наводяться вихідні документи та запити чи програми машинної реалізації.

Завдання підготував
науковий керівник _____

Артемчук Володимир Олександрович

“ _____ ” _____ 2024 р.

Завдання одержав
студент _____

Гончарук Владислав Сергійович

“ _____ ” _____ 2024 р.

АНОТАЦІЯ

кваліфікаційного бакалаврського проекту студента 4 курсу
Навчально-наукового інституту «Інститут інформаційних
технологій в економіці» **Гончарука Владислава
Сергійовича**, виконаної на тему: «Проектування
інформаційної системи обліку будівельних інструментів»
Київ: кафедра інформаційних систем в економіці, 2024 р.

Кваліфікаційний бакалаврський проект присвячений актуальній проблемі автоматизації обліку та контролю за використанням будівельних інструментів на підприємстві, що розглядається з використанням інформаційних технологій. Проект складається з трьох розділів, які логічно пов'язані між собою.

У першому розділі надається характеристика предметної галузі та об'єкта дослідження, проводиться аналіз задач, що розв'язуються, та наводиться перелік існуючих програмних засобів для обліку будівельних інструментів.

Другий розділ є проектним і присвячений обґрунтуванню методів проектування системи, розробці її архітектури, постановці завдань та розробці алгоритмів їх розв'язання.

Третій розділ – конструктивний. Тут наводиться інформаційне, технічне та програмне забезпечення для підсистеми обліку будівельних інструментів. Висвітлені питання організації інформаційного забезпечення: розроблена структура інформаційного забезпечення, описані та наведені форми вхідних та вихідних документів. Обґрунтований комплекс технічних засобів та програмне забезпечення, що використовуються для створення системи обліку будівельних інструментів. Висновки містять рекомендації щодо доцільності розроблення та впровадження системи обліку будівельних інструментів на підприємстві.

РЕФЕРАТ

Кваліфікаційний бакалаврський проект містить 68 сторінок, 5 таблиць, 37 рисунків, список літератури з 7 найменувань, 7 додатків.

Проектування інформаційної системи обліку будівельних інструментів

Перелік ключових слів: управління, контроль, інструмент, база даних, система, сторінка, облік, проект, діаграма, моделювання.

Предметом дослідження є процеси та механізми управління і контролю за використанням будівельних інструментів.

Об'єктом дослідження виступає інформаційна система обліку будівельних інструментів.

Мета бакалаврського проекту полягає в проектуванні та розробці інформаційної системи, яка забезпечує ефективне управління та контроль за використанням будівельних інструментів для підвищення доступності та ефективності їх використання.

Завданнями кваліфікаційного бакалаврського проекту є:

- Дослідження існуючих підходів до управління та контролю за використанням будівельних інструментів.
- Розробка концепції інформаційної системи обліку будівельних інструментів.
- Визначення вимог до системи.
- Проектування та реалізація основних функціональних модулів системи.
- Розробка інформаційної системи для управління та контролю за використанням будівельних інструментів.

Апаратні та програмні засоби, що використовувались на при проектуванні:

- Ноутбук HP - 15-ay108ur.
- Операційна система Windows 10.
- Система керування базами даних MySQL.
- Інтегроване середовище розробки Visual Studio Code для написання коду.

Під час роботи була розроблена інформаційна система для обліку будівельних інструментів. Система включає сторінки списку інструментів, користувачів, об'єктів та повідомлень. Додано можливість відправляти та обробляти запити на інструменти та додавати коментарі. Реалізовано сторінку детальної інформації про інструмент з функціями додавання коментарів, перегляду історії користування та відправлення запитів.

Отримані результати можуть бути використані для полегшення контролю та управління будівельними інструментами на підприємстві. Рекомендується впровадження системи для забезпечення ефективного використання ресурсів та зменшення часу, витраченого на управління інструментами.

Рік виконання кваліфікаційного бакалаврського проекту 2024

Рік захисту кваліфікаційного бакалаврського проекту 2024

ЗМІСТ

Перелік умовних позначень, символів, одиниць виміру та термінів.	3
ВСТУП	4
РОЗДІЛ 1. ХАРАКТЕРИСТИКА ТА АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ	6
1.1. Характеристика предметної галузі та об'єкта дослідження	6
1.2. Аналіз літературних джерел та практичного досвіду використання ІС і технологій в предметній галузі	9
РОЗДІЛ 2. РОЗРОБКА ВИМОГ І МОДЕЛЮВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ	15
2.1. Аналіз і специфікація вимог до інформаційної системи	15
2.2. Постановка задачі	17
2.2.1. Характеристика задачі	17
2.2.2. Вихідна інформація	23
2.2.3. Вхідна інформація	24
2.2.4. Алгоритм розв'язання задачі	25
2.2.5. Математичний опис	25
2.3. Моделювання інформаційної системи/підсистеми	26
2.3.1. Моделювання поведінки системи	26
2.3.2. Розподіл вимог за компонентами системи	33
РОЗДІЛ 3. ПРОЕКТУВАННЯ ТА РЕАЛІЗАЦІЯ КОМПОНЕНТІВ СИСТЕМИ	37
3.1. Інформаційне забезпечення	37
3.1.1. Загальна характеристика інформаційного забезпечення	37
3.1.2. Організація збору і передавання первинної інформації	37
3.1.3. Побудова системи класифікації та кодування	39
3.1.4. Проектування форм первинних документів	41
3.1.5. Вибір СКБД	44
3.1.6. Інфологічна модель бази (сховища) даних	46
3.1.7. Даталогічна модель бази (сховища) даних	48
3.2. Технічне забезпечення	51
3.3. Програмне забезпечення	54
3.4. Результати реалізації інформаційної системи	56
ВИСНОВКИ	65
ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ	68
ДОДАТКИ	69

Перелік умовних позначень, символів, одиниць виміру та термінів.

ІС - інформаційна система;

ЕОМ - електронно-обчислювальна машина;

АРМ - автоматизоване робоче місце;

БД - база даних;

СУБД - система управління базами даних;

LAN - локальна мережа (Local Area Network);

API – інтерфейс програмування додатків (Application Programming Interface);

ПЗ – програмне забезпечення;

ВСТУП

У сучасних умовах розвитку будівельної галузі, автоматизація процесів обліку та управління інструментами є важливою складовою ефективної роботи підприємств. З кожним роком кількість інструментів, що використовуються на будівельних майданчиках, зростає, що ускладнює їх облік та контроль. Належний облік та управління інструментами дозволяють значно зменшити втрати, підвищити продуктивність праці та забезпечити безперебійний робочий процес.

Актуальність даного дослідження обумовлена необхідністю вирішення проблеми автоматизації процесів обліку інструментів на будівельних підприємствах. Існуючі на ринку рішення не завжди задовольняють специфічні потреби підприємств, що веде до необхідності розробки нових, більш ефективних та адаптованих систем.

Метою даного кваліфікаційного бакалаврського проекту є розробка інформаційної системи обліку будівельних інструментів, яка дозволить автоматизувати процеси обліку, моніторингу та управління інструментами, підвищити точність обліку та зменшити витрати часу на обробку даних.

Об'єктом дослідження є процеси обліку та управління будівельними інструментами на підприємстві. Предметом дослідження виступає інформаційна система, яка забезпечує автоматизацію цих процесів.

Для досягнення мети проекту були поставлені наступні завдання:

1. Аналіз існуючих систем обліку та управління інструментами.
2. Розробка архітектури інформаційної системи обліку інструментів.
3. Реалізація основних функціональних модулів системи: облік інструментів, управління запитами на використання інструментів, моніторинг та сповіщення користувачів.
4. Тестування розробленої системи та оцінка її ефективності.

В ході виконання проекту були використані наступні методи дослідження: аналіз літературних джерел, системний аналіз, структурно-логічний метод, метод моделювання.

Огляд літератури показав, що наукова спільнота активно досліджує питання автоматизації обліку та управління ресурсами на підприємствах. Існує велика кількість публікацій, що розглядають різні аспекти цієї проблеми, проте більшість з них зосереджені на теоретичних аспектах або описують загальні принципи побудови систем управління. Огляд найбільш актуальних робіт дозволив визначити основні напрямки розробки та вимоги до інформаційної системи обліку інструментів.

Теоретична значущість даної роботи полягає у розробці нових підходів до автоматизації процесів обліку та управління інструментами на будівельних підприємствах. Практична значущість полягає у створенні програмного продукту, який може бути використаний для підвищення ефективності роботи підприємств та зниження витрат, пов'язаних з втратою та нераціональним використанням інструментів.

Структура роботи включає вступ, основну частину, що складається з теоретичних та практичних розділів, та висновки. У вступі обґрунтовано вибір теми дослідження, визначено об'єкт та предмет дослідження, сформульовано мету та завдання проекту. Основна частина містить детальний опис розробленої інформаційної системи, включаючи аналіз існуючих рішень, розробку архітектури системи, реалізацію основних функціональних модулів та результати тестування. У висновках підведено підсумки виконаної роботи, оцінено її ефективність та запропоновано рекомендації щодо подальшого вдосконалення системи.

Таким чином, кваліфікаційний бакалаврський проект спрямований на вирішення актуальної проблеми автоматизації обліку та управління будівельними інструментами, що дозволить підвищити ефективність роботи підприємств та забезпечити їх стабільний розвиток.

РОЗДІЛ 1. ХАРАКТЕРИСТИКА ТА АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1. Характеристика предметної галузі та об'єкта дослідження

Предметною галуззю є управління та контроль за використанням будівельних інструментів під час будівельних робіт. Головною метою цієї галузі є забезпечення належного рівня доступності та ефективного використання будівельних інструментів для здійснення різних будівельних проектів на підприємстві.

Об'єктом дослідження є інформаційна система обліку будівельних інструментів. Ця система спрямована на забезпечення ефективного управління та контролю за використанням будівельних інструментів на підприємстві. Вона включає в себе комплекс програмних і апаратних засобів, які дозволяють здійснювати облік, видачу та повернення інструментів, автоматизувати процеси зберігання та розподілу інструментів між працівниками підприємства.

Інформаційна система обліку будівельних інструментів спрямована на оптимізацію процесів управління цими ресурсами, забезпеченням їх ефективного використання та запобігання можливих втрат чи непорозумінь стосовно їх доступності та стану.

Будівельне підприємство це основний суб'єкт управління та контролю за використанням будівельних інструментів. Воно відповідає за реалізацію будівельних проектів та забезпечення належного рівня доступності і ефективного використання необхідних інструментів.

Об'єктами будівництва є конкретні місця, де проводяться будівельні роботи. Ці об'єкти можуть бути різного типу: від житлових будинків до комерційних об'єктів. Інформація про ці об'єкти зберігається в інформаційній системі обліку будівельних інструментів і використовується для планування та координації використання інструментів.

Складські приміщення: Місця, де зберігаються будівельні інструменти до їх використання. Ці об'єкти можуть включати в себе склади на території підприємства, а також орендовані або підпільні склади. Ефективне управління

цими приміщеннями важливе для забезпечення належного рівня доступності інструментів та попередження втрат.

Щодо користувачів системи, існує дві категорії:

- Будівельники: Професійні робітники, які виконують різноманітні будівельні завдання на об'єктах будівництва. Вони використовують будівельні інструменти для виконання своїх обов'язків і є основними кінцевими користувачами системи.
- Працівники складу: Це особи, відповідальні за прийом, зберігання та видачу будівельних інструментів на складі. Вони відіграють важливу роль у забезпеченні належного управління запасами та контролю за їх використанням.

У предметній галузі управління та контролю за використанням будівельних інструментів на підприємстві приймаються рішення, що мають сутність проблемних ситуацій та спрямовані на досягнення цілей оптимізації процесів управління цими ресурсами. Основні рішення, що приймаються, стосуються обробки запитів щодо видачі та повернення будівельних інструментів.

У ситуації, коли користувач подає запит на повернення інструменту, працівник складу здійснює перевірку інструменту та приймає рішення про прийняття чи відхилення запиту. Якщо запит на повернення надійшов, але інструмент не був переданий працівнику складу, запит буде відхилений. Однак, у випадку, якщо користувач принесе інструмент і він знаходиться в задовільному стані, запит приймається. У випадку, коли інструмент виявлено з відмовою, його статус буде змінено на "виведено з експлуатації".

Також важливим є рішення користувачів щодо відправлення запитів на отримання інструментів, які їм потрібні. Якщо користувач надсилає запит, знаходячись поза складом, цей запит буде відхилено, оскільки відсутній отримувач інструмента.

Особи, що приймають рішення в цій предметній галузі, включають працівників складу, керівництво підприємства, а також користувачів системи. Функції їх переваг полягають у вирішенні конкретних проблемних ситуацій та

досягненні цілей оптимізації процесів управління інструментами. Принципи узгодження індивідуальних переваг у разі групового прийняття рішень включають врахування потреб кожного учасника процесу та пошук компромісних рішень, які задовольняють всіх зацікавлених сторін.

На середовище прийняття рішень впливають різноманітні фактори, такі як технічні можливості, фінансові обмеження, потреби користувачів та зовнішні впливи. Ці фактори визначають характер і ступінь впливу на процес прийняття рішень, а також сприяють формуванню стратегій управління та контролю за використанням будівельних інструментів.

Взаємозв'язки рішень в цій галузі полягають у тому, що кожне прийняте рішення впливає на подальші рішення та може мати наслідки для майбутніх процесів управління інструментами. Таким чином, необхідно враховувати попередні рішення та їх вплив на майбутні вирішення завдань.

Щоб прийняти обґрунтовані рішення, необхідна внутрішня та зовнішня інформація. Вона може бути на основі досвіду або об'єктивною та надходити з бази даних підприємства.

У ситуаціях прийняття рішень можна виділити різні типи та види ситуацій. До них відносяться ситуації закритих задач, коли відомі всі умови та альтернативи, ситуації відкритих задач, коли умови не повністю визначені, та кризові ситуації, коли потрібно негайно реагувати на непередбачені обставини. Кожен тип ситуації вимагає відповідного підходу та стратегії прийняття рішень, що дозволяє ефективно управляти процесами в умовах невизначеності та складності.

Отже, управління та контроль за використанням будівельних інструментів на підприємстві вимагає систематичного аналізу, прийняття обґрунтованих рішень та впровадження ефективних стратегій управління ресурсами.

1.2. Аналіз літературних джерел та практичного досвіду використання ІС і технологій в предметній галузі

Під час аналізу літературних джерел та практичного досвіду використання інформаційних систем (ІС) та технологій у сфері обліку будівельних інструментів виявлено цілий ряд підходів та програмних рішень, спрямованих на покращення ефективності управління цими активами.

Зокрема, було звернено увагу на те, що деякі існуючі програми не завжди відповідають конкретним потребам у сфері обліку будівельних інструментів. Замість цього, вони частіше зорієнтовані на загальні завдання бізнесу, надаючи широкий спектр можливостей для управління виробничим процесом та ресурсами в цілому. Ці інтегровані системи охоплюють такі аспекти, як фінансовий облік, управління проектами, ресурсами та звітність, створюючи таким чином цілісну платформу для ефективного керування виробничими операціями підприємства.

Важливо відзначити, що такий глобальний підхід може бути корисним для підприємств, оскільки він дозволяє не лише здійснювати точний облік інструментів, але й оптимізувати всю бізнес-діяльність та управління ресурсами в цілому. Втім, наявність деяких недоліків у таких програмах також варто враховувати.

Зокрема, виявлено, що більшість програм мають високу вартість, а також обмеження на кількість користувачів та обсяг доданих інструментів. Крім того, велика кількість функцій може ускладнити процес їх використання та вимагати додаткового навчання персоналу, що призводить до додаткових витрат на підготовку та підтримку системи.

Додатково до цього, складність використання програмних рішень може призвести до зростання витрат на підтримку системи. Компанії змушені виділяти додаткові ресурси на технічну підтримку та обслуговування програмного забезпечення, щоб забезпечити його безперебійну роботу.

Такі обмеження та складнощі можуть стати серйозними перешкодами для ефективного впровадження програмних рішень у практику будівельної галузі.

Характеристики розглянутих нами існуючих програмних рішень наведено у табл. 1.1.

Таблиця 1.1.

Характеристики існуючих програмних рішень.

Характеристика	Archdesk	Xero	Sage 50	QuickBooks
Ціна	Від 790\$/місяць	Від 29\$ до 62\$/місяць	Від 595\$/рік за одного користувача	Від 1464\$/рік
Обмеження кількості користувачів/інструментів	Обмежено для версії за 790\$/місяць	Обмежено	Обмежено	Обмежено
Українська мова	Ні	Ні	Ні	Ні
Управління проектами	Так	Так	Так	Ні
Контроль фінансів	Так	Так	Так	Так
Простота використання	Інтуїтивно зрозумілий інтерфейс	Потрібно проходити курс	Через велику кількість функцій складно розібратись з інтерфейсом	Може знадобитись час щоб навчитись працювати з сервісом
Технологічно-організаційні аспекти	Система спрямована в першу чергу на планування завдань	Система спрямована на бухгалтерський облік	Система розподілена на сервіси для застосування в різних сферах	Система спрямована на керування бізнесом

Детальніше розглянемо програмні рішення від українських компаній.

Ardix FastID Tools.

Програмне забезпечення Ardix FastID Tools створено з метою упорядкування процесу збирання, видачі та повернення інструментів і обладнання, а також для систематичного контролю та інвентаризації цього обладнання.

Ця система використовує RFID мітки для ідентифікації кожного інструменту, а також читачі для збору даних. Ardix FastID Tools дозволяє отримувати повну звітність щодо використання інструментів за кілька секунд: хто, коли та який інструмент отримав, його стан, кількість інструментів, що перебувають у співробітників або клієнтів, та інші аспекти.

RFID система обліку інструменту Ardix FastID Tools знаходить своє застосування в різних галузях, таких як компанії, які надають обладнання в оренду, великі виробничі підприємства, IT компанії, сільське господарство, державні установи, фінансовий сектор, морські судна, а також компанії, що здійснюють ремонт обладнання на місці обслуговування клієнтів.

Програма обліку видачі інструментів Ardix FastID Tools є гнучкою і може використовуватися як самостійний інструмент для управління обладнанням, так і інтегрована в основну систему управління. Комплект Ardix FastID Tools включає веб-інтерфейс для користувача, мобільний додаток на RFID терміналі та серверну частину у вигляді rest сервісу.

Серед переваг Ardix FastID Tools варто відзначити можливість розгортання як сервісу на стороні замовника (на власному сервері або в хмарному сервісі), зрозумілий та дружній до користувача інтерфейс та зручний мобільний додаток.

Вартість ПЗ ArdixID Tools може бути варіативною залежно від обраної опції. Детальнішу інформацію наведено у табл. 1.2.

Таблиця 1.2.

Вартість ПЗ ArdixID Tools.

Підписка	ПЗ для ручного зчитувача	\$15/міс.
	Повне ПЗ	\$80/міс.
Купівля	ПЗ для ручного зчитувача	\$300
	Повне ПЗ	\$2400
Додатково	Установка та настройка ПЗ	\$800
	Підтримка ПЗ	\$30/міс.

EqMan.

Програмне забезпечення "EqMan" створено з метою забезпечення ефективного обліку та контролю руху інструментів та обладнання на

будівельних майданчиках. Ця система дозволяє вести детальний облік переміщень кожного інструменту, зберігаючи історію операцій з ним, та надає доступ до повної інформації про місцеперебування, кількість та стан інструменту в будь-який момент часу. Однією з ключових переваг програми "EqMan" є забезпечення прозорого обліку інструментів та обладнання, що сприяє ефективній комунікації між робітниками на будівельних об'єктах. Крім того, завдяки миттєвому пошуку за допомогою QR-кодів, користувачі можуть швидко локалізувати необхідний інструмент і отримати докладну інформацію про його характеристики та історію використання.

Додатково, програма "EqMan" забезпечує високий рівень безпеки за рахунок подвійної верифікації операцій з товарно-матеріальними цінностями, що допомагає уникнути можливих випадків шахрайства та крадіжок на будівельних майданчиках.

Щодо недоліків програми "EqMan", до них можна віднести обмежені можливості інтеграції з існуючими системами управління обладнанням, високу вартість програми та послуг її установки та підтримки, а також необхідність наявності технічних знань для ефективного використання програмного забезпечення. У порівнянні з програмою "Ardix FastID Tools", "EqMan" відрізняється більш широким спектром функцій та більшим рівнем зручності в користуванні. Хоча обидві програми спрямовані на досягнення подібних цілей, "EqMan" може бути більш привабливим варіантом для користувачів, які цінують простоту використання та розширені можливості управління будівельним обладнанням.

Однак, Ardix FastID Tools має свої переваги, такі як більша гнучкість у налаштуванні та можливість інтеграції з іншими системами управління, що може зробити його більш привабливим для певних сегментів користувачів.

Інформацію про підписки на EqMan наведено у табл. 1.3.

Таблиця 1.3.

Підписки EqMan.

Тип підписки	Ціна (грн./рік)	Кількість інструментів	Кількість користувачів
Free	0	50	1
Optimal	12048	400	40
Optimal+	24516	2500	100
Premium	41580	Без ліміту	Без ліміту

Тема інформаційної системи обліку будівельних інструментів є надзвичайно актуальною в сучасному світі будівництва та управління проектами. Будівництво - це складний та великий процес, який вимагає ефективного управління ресурсами, включаючи інструменти. На сьогоднішній день, коли технології швидко розвиваються, використання інформаційних систем для автоматизації цього процесу стає важливою потребою.

Перш за все, система обліку інструментів допомагає зберігати точну інформацію про наявність та розподіл інструментів на будівельному майданчику. Це зменшує ризик втрати інструментів, що може призвести до затримок у роботі та додаткових фінансових витрат. Така система дозволяє миттєво отримувати дані про місцезнаходження кожного інструменту, його стан та відповідальних осіб, що сприяє більш ефективному плануванню робочих процесів.

По-друге, з використанням інформаційної системи можна відслідковувати використання інструментів кожним працівником. Це дозволяє підтримувати дисципліну та відповідальність серед персоналу, а також уникнути випадків викрадення чи недбалого використання інструментів. Інформаційні системи забезпечують прозорість і контроль над кожним інструментом, що знижує ймовірність несанкціонованого використання або втрати.

Ще однією важливою перевагою такої системи є її спроможність попереджати витрати через втрату інструментів. Правильне відслідковування

та управління інвентарем може зменшити необхідність у поновленні втрачених інструментів, що, в свою чергу, знижує загальні витрати на будівельний проект.

Хоча існують вже програмні рішення для обліку інструментів, проте розробка більш спеціалізованої системи, яка буде більш вузько направлена на потреби будівельної галузі, може мати значний попит. Спеціалізовані системи можуть враховувати особливості різних будівельних процесів, типи інструментів та матеріалів, що використовуються, і навіть специфіку окремих проектів. Простий та зрозумілий інтерфейс такої системи робить її доступною для широкого кола користувачів, включаючи тих, хто не має глибоких знань інформаційних технологій. Це дозволяє швидко інтегрувати систему у робочий процес, мінімізуючи час на навчання та адаптацію персоналу.

Важливо зазначити, що успішне впровадження таких систем може суттєво підвищити ефективність управління ресурсами, знизити витрати та покращити загальну продуктивність будівельних проектів. Спеціалізовані інформаційні системи можуть стати ключовим елементом у забезпеченні конкурентоспроможності компаній у будівельній галузі.

Отже, тема інформаційної системи обліку будівельних інструментів є актуальною через потребу в ефективному управлінні ресурсами та зменшенні витрат у будівельній галузі. Впровадження такої системи може сприяти підвищенню продуктивності, зниженню ризиків та покращенню загальної ефективності будівельних проектів.

РОЗДІЛ 2. РОЗРОБКА ВИМОГ І МОДЕЛЮВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

2.1. Аналіз і специфікація вимог до інформаційної системи

Система повинна забезпечувати швидкий та легкий доступ до актуальної інформації про наявність і стан будівельних інструментів для всіх зацікавлених сторін в організації.

- Система повинна забезпечувати облік користувачів відповідальних за інструменти.
- Система повинна забезпечувати захищений доступ до інформації про будівельні інструменти та обмежувати права доступу до цієї інформації відповідно до ролей та обов'язків користувачів.
- Система повинна бути доступною з різних пристроїв та платформ, щоб користувачі могли отримувати доступ до неї незалежно від місця перебування.
- Система може фіксувати переміщення інструментів між робочими місцями, складами, проектами ведучи журнал видачі та повернення інструментів.
- Система повинна допомагати забезпечувати безпеку працівників шляхом відслідковування відповідності будівельних інструментів стандартам безпеки та технічного стану.
- Система повинна забезпечувати повний та точний облік всіх будівельних інструментів, включаючи їхнє розташування, стан, історію використання.

Діаграму функціональних вимог наведено на рисунку 2.2.

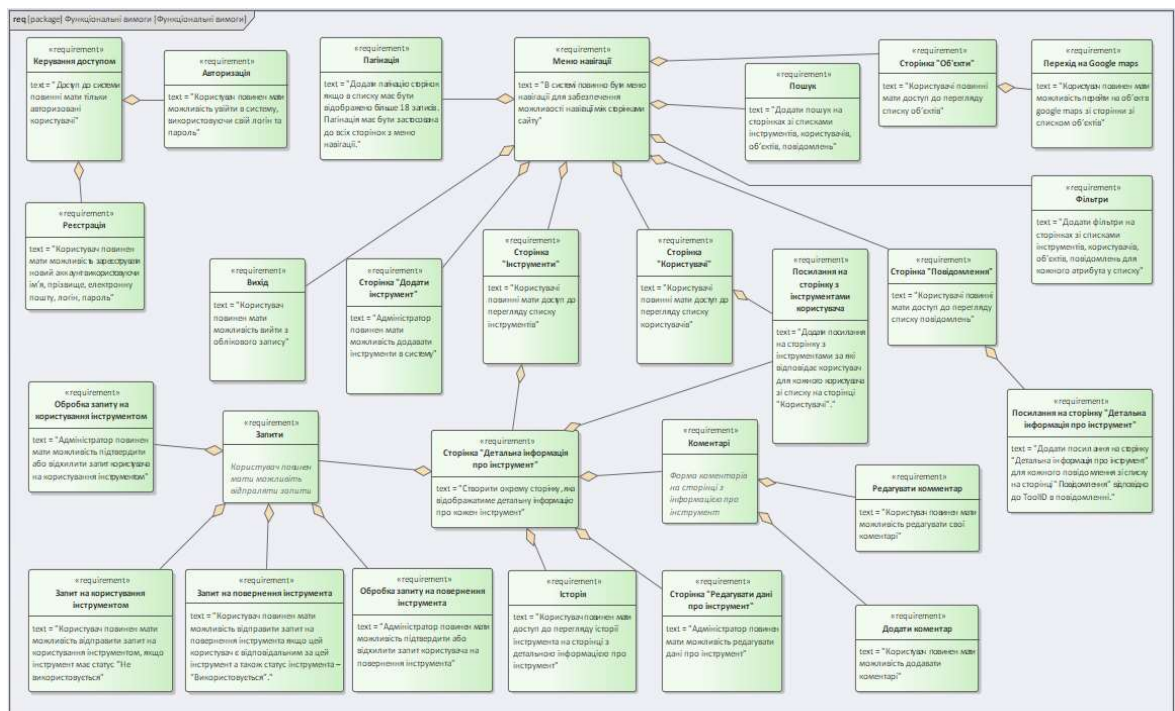


Рисунок 2.2. Діаграма функціональних вимог.

Діаграму нефункціональних вимог наведено на рисунку 2.3.

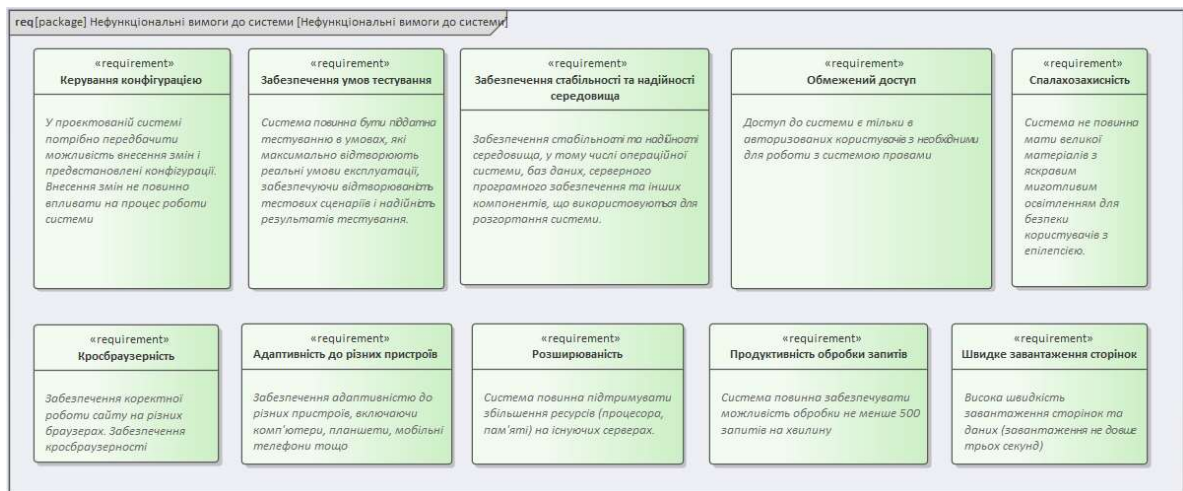


Рисунок 2.3. Діаграма нефункціональних вимог.

Деталізовані специфікації вимог наведено у додатку 1.

2.2. Постановка задачі

2.2.1. Характеристика задачі

Призначення та техніко-економічна сутність задачі.

Призначення інформаційної системи обліку будівельних інструментів полягає в забезпеченні ефективного та систематизованого контролю за використанням та розподілом інструментів на будівельному об'єкті. Основна техніко-економічна сутність цієї задачі полягає в оптимізації процесів управління інвентарем, що включає в себе ведення обліку, моніторинг та забезпечення доступності необхідного обладнання для будівельних робіт.

Необхідність розв'язання даної задачі за допомогою електронно-обчислювальних машин (ЕОМ) визначається потребою у високій швидкості обробки великого обсягу даних, точності обліку та автоматизації процесів контролю. Така система дозволить значно зменшити ймовірність помилок у веденні обліку, оптимізувати час, необхідний для проведення інвентаризацій, а також забезпечить швидкий доступ до інформації для прийняття управлінських рішень.

Обґрунтування потреби розв'язання за допомогою ЕОМ.

В умовах сучасного будівельного господарства об'єкти можуть бути великими та складними, що ускладнює ведення їх обліку та контролю. Використання традиційних методів обліку, які базуються на ручній роботі та

паперовій документації, стає неефективним та часомістким. Отже, розв'язання цієї задачі за допомогою ЕОМ є необхідним для підвищення ефективності управління та зменшення ймовірності помилок.

ЕОМ дозволить автоматизувати процеси обліку, забезпечить швидкий доступ до інформації про наявність і розподіл інструментів, а також спростить ведення документації. Це забезпечить підвищення продуктивності праці та зменшення витрат часу на вирішення рутинних адміністративних завдань.

Розв'язання задачі інформаційної системи обліку будівельних інструментів передбачає управління різноманітними об'єктами та процесами, пов'язаними з будівельними ресурсами. Перелік об'єктів, за управлінням якими розв'язується задача, включає:

- Інструменти: Це основний об'єкт управління, який включає в себе перелік доступних інструментів, їх стан, місцезнаходження та історію використання.
- Користувачі: Інформаційна система повинна вести облік користувачів, які мають доступ до будівельних інструментів, включаючи їхні дані про авторизацію та права доступу.
- Об'єкти будівництва: До цієї категорії входять будівельні об'єкти, на яких використовуються будівельні інструменти. Система повинна вести облік цих об'єктів та забезпечувати доступ до інструментів для відповідних будівельних проектів.
- Запити на користування інструментом: Інформаційна система повинна враховувати запити користувачів на позичення або повернення будівельних інструментів.
- Коментарі та повідомлення: До об'єктів управління також входять коментарі користувачів та системні повідомлення, які можуть містити важливу інформацію про стан та використання інструментів.

Розв'язання задачі включає в себе ефективне управління цими об'єктами з метою забезпечення оптимального використання ресурсів, підвищення продуктивності та ефективності будівельних процесів.

Призначення та використання вихідної інформації в інформаційній системі обліку будівельних інструментів має на меті забезпечення оперативного та зручного доступу до необхідної інформації для користувачів системи. Вихідна інформація генерується системою на основі обробки вхідних даних та забезпечує користувачів актуальною та корисною інформацією. Нижче подано опис призначення та використання основних видів вихідної інформації:

- **Інформація про інструменти (Tools_info):** Ця вихідна інформація містить перелік доступних будівельних інструментів разом з їхніми характеристиками та станом. Вона призначена для надання користувачам повного огляду доступного інвентарю та його параметрів для вибору необхідного інструменту для конкретного завдання.
- **Список коментарів (Comments_list):** Ця вихідна інформація містить коментарі та відгуки користувачів щодо використання конкретних інструментів. Вона допомагає іншим користувачам оцінити якість та ефективність інструментів, а також надає можливість для зворотного зв'язку і вдосконалення якості обслуговування.
- **Історія користування інструментом (Tool_history):** Ця вихідна інформація відображає історію використання конкретного інструменту, включаючи дати, час та користувачів, які користувалися ним. Вона допомагає відстежувати історію використання та підтримувати належний контроль за інвентарем.
- **Список об'єктів (Object_list):** Ця вихідна інформація містить перелік будівельних об'єктів, на яких використовуються будівельні інструменти. Вона дозволяє користувачам відстежувати, де саме використовуються конкретні інструменти, та забезпечує керівництву інформацію для прийняття управлінських рішень.
- **Список повідомлень (Notification_list):** Ця вихідна інформація містить повідомлення системи, такі як сповіщення про планові та непланові події, оновлення програмного забезпечення тощо. Вона дозволяє

користувачам та адміністраторам системи отримувати актуальну інформацію про стан системи та події, що відбуваються.

- Інформація про користувачів (Users_info): Призначення та використання вихідної інформації про користувачів полягає у забезпеченні системи актуальною та повною інформацією про користувачів, які мають доступ до системи обліку будівельних інструментів. Ця інформація дозволяє здійснювати контроль за доступом до інструментів, а також забезпечує можливість відстеження діяльності користувачів у системі.

Періодичність розв'язання задачі та видачі вихідної інформації визначається як миттєва, бо надається не довше ніж через 5 секунд, при стабільному підключенні до інтернет мережі. Інформація надається користувачам системи безпосередньо в момент її запиту або після виникнення певної події, що спричиняє генерацію вихідної інформації. Такий підхід забезпечує максимальну актуальність та своєчасність надання інформації, що дозволяє користувачам оперативно приймати рішення та виконувати необхідні дії в системі.

Автоматизоване розв'язання задачі може припинятися з різних причин. Одна з них - технічні проблеми, що виникають у програмному забезпеченні або обчислювальному обладнанні. Такі проблеми можуть призвести до недоступності системи для користувачів і, отже, призупинення автоматизованого процесу розв'язання завдань.

Планове технічне обслуговування є ще однією причиною припинення автоматизованого розв'язання. Це може включати в себе оновлення програмного забезпечення, вдосконалення функціоналу або зміни конфігурації системи. Під час такого обслуговування система може бути тимчасово недоступною для користувачів.

Додатково, адміністративні рішення можуть також призвести до припинення автоматизованого розв'язання задачі. Наприклад, у зв'язку зі зміною стратегії підприємства або вимогами регуляторів, може бути необхідно внести зміни у роботу системи, що може призвести до тимчасового зупинення автоматизованого процесу.

Крім того, неможливість автоматизувати повністю деякі аспекти роботи системи, які потребують участі людей, також може спричинити припинення автоматизованого розв'язання. Наприклад, процеси перевірки, видачі, отримання та повернення інструментів потребують присутності людей для виконання відповідних дій, що не можуть бути повністю автоматизованими. Отже, це фактори можуть призвести до припинення автоматизованого розв'язання задачі в інформаційній системі обліку будівельних інструментів.

Ця задача має певний зв'язок зі сторонньою інформаційною системою для вирішення наших задач. Конкретно, існує можливість переходу зі сторінки об'єктів на Google Maps, де користувач може переглянути список складів та об'єктів будівництва. Це створює зручний інтерфейс для користувачів, які можуть швидко отримати доступ до локацій будівництва та складських приміщень безпосередньо на мапі.

За винятком цього переходу, інформаційна система обліку будівельних інструментів працює автономно і не має прямих зв'язків з іншими системами. Вона надає комплексні рішення лише у межах своєї функціональності, спрощуючи процес управління інструментами та обліку ресурсів для користувачів.

У процесі взаємодії персоналу із технічними засобами в рамках розв'язання завдань з управління будівельними інструментами розподіл дій залежить від конкретного контексту. Персонал системи, використовуючи інтерфейс програмного забезпечення, здійснює ряд операцій, які включають у себе пошук необхідних інструментів у базі даних, контроль їх доступності, видачу та повернення користувачам. Ці дії вимагають від працівників систематичного моніторингу та взаємодії з інтерфейсом системи.

Технічні засоби, у свою чергу, забезпечують автоматизований доступ до бази даних, що дозволяє оперативно оновлювати та відображати актуальну інформацію про наявність інструментів. Вони також автоматизують процеси реєстрації видачі та повернення інструментів, спрощуючи рутинні завдання персоналу.

Крім того, персонал системи використовує технічні засоби для управління базою даних, де здійснюється додавання нових записів про інструменти, їх редагування та видалення. Це вимагає від працівників системи ретельного контролю та обробки інформації за допомогою інтерфейсу програмного забезпечення.

Таким чином, взаємодія персоналу із технічними засобами в інформаційній системі обліку будівельних інструментів є важливим аспектом забезпечення ефективності та продуктивності роботи системи. Вона спрощує процес управління інструментами та дозволяє оперативно реагувати на потреби користувачів.

Інформаційна модель задачі представлена на рисунку 2.4.

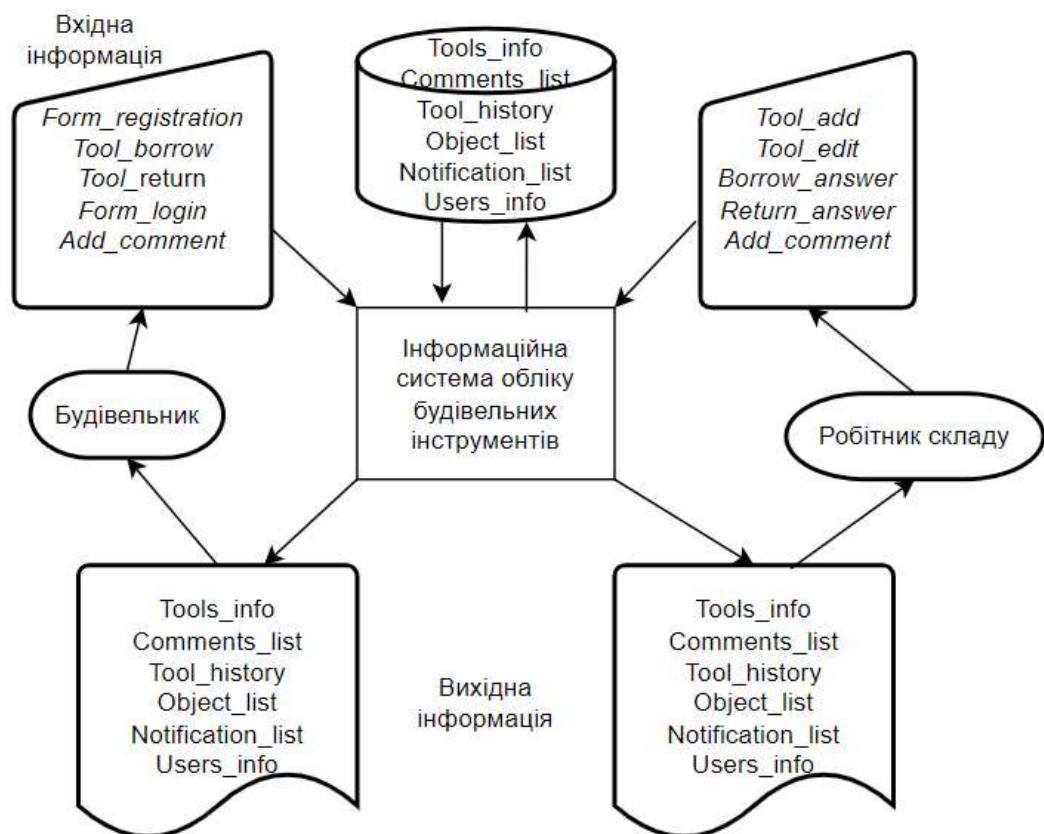


Рисунок 2.4. Інформаційна модель задачі.

2.2.2. Вихідна інформація

Перелік і опис вихідних повідомлень наведено у табл. 2.1.

Таблиця 2.1.

Перелік та опис вихідних повідомлень.

Назва вихідного повідомлення	Ідентифікатор	Форма подання і вимоги до неї	Періодичність видання	Термін видання і допустимий час затримки	Користувач інформації
Інформація про інструменти	Tools_info	Екранна форма	При переході користувача на сторінку з списком інструментів	Одразу після переходу на сторінку(затримка то 5 секунд)	Користувачі системи
Список коментарів	Comments_list	Екранна форма	При переході користувача на сторінку детальної інформації про інструмент	Одразу після переходу на сторінку(затримка то 5 секунд)	Користувачі системи
Історія користування інструментом	Tool_history	Екранна форма	При переході користувача на сторінку детальної інформації про інструмент	Одразу після переходу на сторінку(затримка то 5 секунд)	Користувачі системи
Список об'єктів	Object_list	Екранна форма	При переході користувача на сторінку списку об'єктів	Одразу після переходу на сторінку(затримка то 5 секунд)	Користувачі системи
Список повідомлень	Notification_list	Екранна форма	При переході користувача на сторінку повідомлень	Одразу після переходу на сторінку(затримка то 5 секунд)	Користувачі системи
Список користувачів	Users_info	Екранна форма	При переході користувача на сторінку з списком користувачів системи	Одразу після переходу на сторінку(затримка то 5 секунд)	Користувачі системи

2.2.3. Вхідна інформація

Вхідною інформацією є дані з форми реєстрації користувача, дані про нові інструменти, запити користувачів на користування інструментом.

Перелік і опис вхідних повідомлень наведено у таблиці 2.2.

Таблиця 2.2.

Перелік та опис вхідних повідомлень.

<i>Назва вхідного повідомлення</i>	<i>Ідентифікатор</i>	<i>Форма подання</i>	<i>Термін і частота надходження</i>	<i>Джерело</i>
Форма реєстрації користувача	<i>Form_registration</i>	Екранна форма	По мірі реєстрації користувачів	Дані від користувача системи
Запит на позичення інструменту	<i>Tool_borrow</i>	Екранна форма	По мірі надсилання запиту на позичення інструмента від будівельника	Запит від будівельника
Запит на повернення інструменту	<i>Tool_return</i>	Екранна форма	По мірі надсилання запиту на повернення інструмента від будівельника	Запит від будівельника
Додавання нового інструменту	<i>Tool_add</i>	Екранна форма	По мірі додавання нових інструментів	Дані від робітника складу
Редагування даних про інструмент	<i>Tool_edit</i>	Екранна форма	По мірі потреби оновлення даних про інструмент	Дані від робітника складу
Форма авторизації користувача	<i>Form_login</i>	Екранна форма	По мірі авторизації існуючих користувачів	Дані від користувача
Відповідь на запит користування інструментом	<i>Borrow_answer</i>	Екранна форма	По мірі надходження відповіді	Відповідь робітника складу
Відповідь на запит повернення інструмента	<i>Return_answer</i>	Екранна форма	По мірі надходження відповіді	Відповідь робітника складу
Додати коментар	<i>Add_comment</i>	Екранна форма	По мірі додавання коментарів	Дані від користувача системи

2.2.4. Алгоритм розв'язання задачі

Алгоритм розв'язання задачі показано на рисунку 2.5.

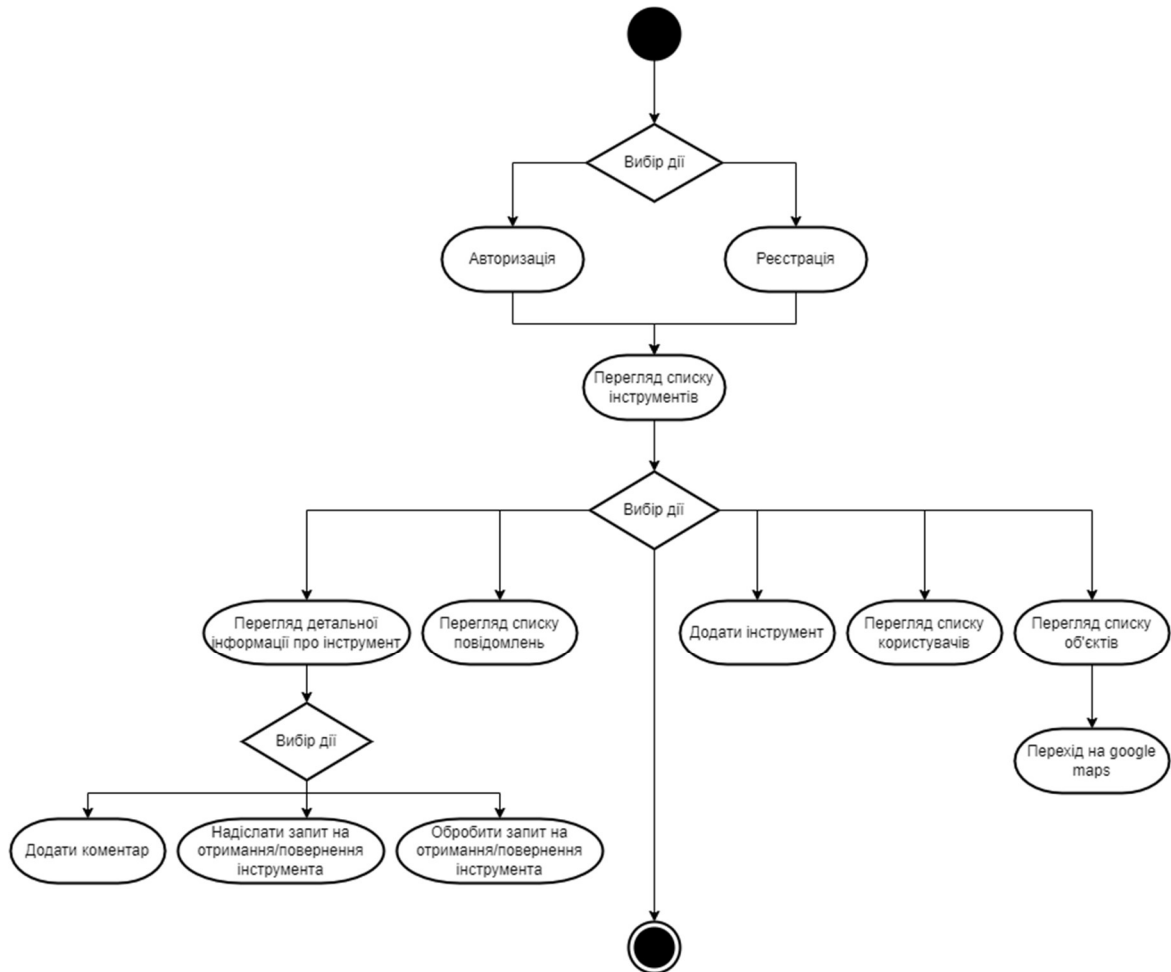


Рисунок 2.5. Алгоритм розв'язання задачі.

2.2.5. Математичний опис

Загальна вартість інструментів на складі:

$$T = \sum_{i=1}^n (Ci \times Qi) \quad (1)$$

де Ci - ціна одиниці інструмента i , Qi - кількість інструментів i , n - загальна кількість різних типів інструментів на складі.

Загальна вартість позичених інструментів:

$$B = \sum_{i=1}^n (Ci \times Li) \quad (2)$$

де Ci - ціна одиниці інструмента i , Li - кількість позичених інструментів i , n - загальна кількість різних типів позичених інструментів.

Загальна кількість інструментів на складі:

$$K = \sum_{i=1}^n Si \quad (3)$$

де Si - кількість інструментів i на складі.

2.3. Моделювання інформаційної системи/підсистеми

2.3.1. Моделювання поведінки системи

У контексті нашої інформаційної системи, акторами є дві основні категорії користувачів: будівельники та працівники складу. Кожна з цих категорій має свої власні ролі та функціональні можливості в системі.

Будівельники виступають у ролі звичайних користувачів і мають обмежений доступ до функціоналу системи. Їхні можливості обмежуються переважно отриманням інформації про наявність інструментів, поданням запитів на позичення чи повернення інструментів, а також додаванням коментарів до певних інструментів.

Працівники складу, натомість, володіють розширеними правами адміністратора. Вони мають повний доступ до функціоналу системи, зокрема можуть керувати базою даних, додавати нові інструменти, редагувати інформацію про них та здійснювати інші адміністративні операції.

Для кращого розуміння ролей та можливостей кожного з цих користувачів була розроблена діаграма прецедентів, яка відображає їхні взаємодії з системою. Ця діаграма, представлена на рисунку 2.6, надає загальний огляд функціональних вимог системи та ілюструє різницю у можливостях між будівельниками та працівниками складу.

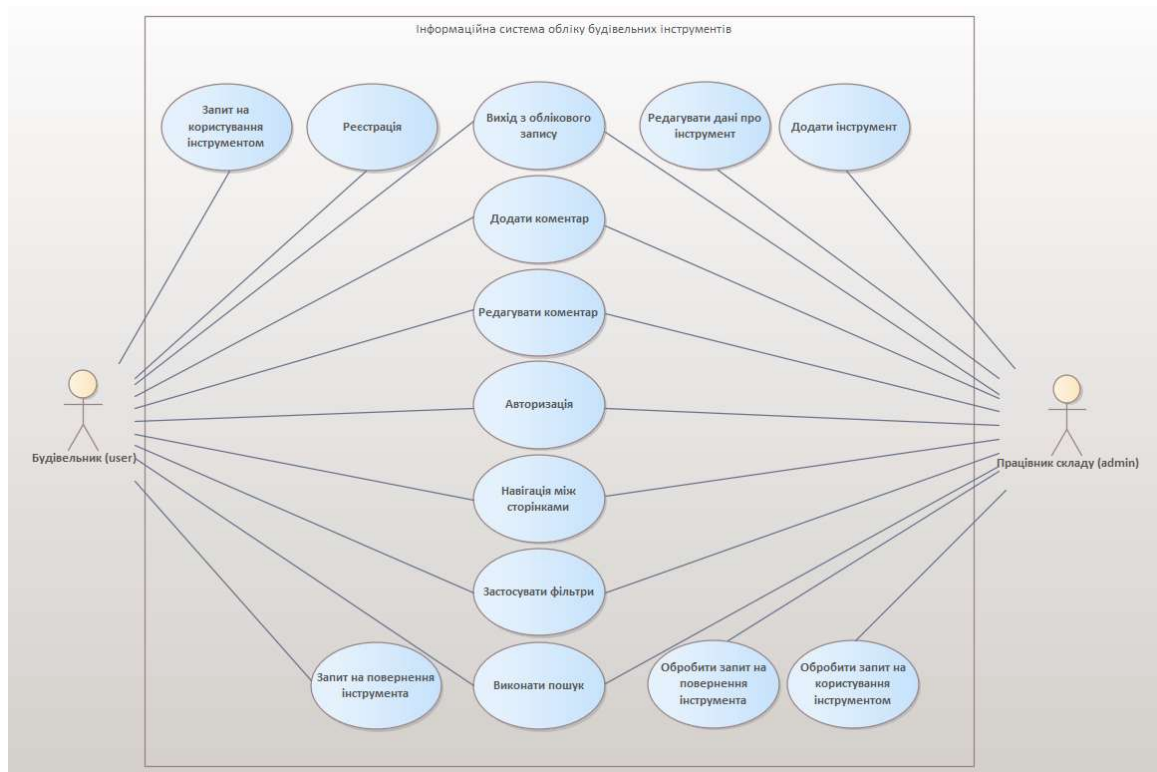


Рисунок 2.6. Діаграма прецедентів.

Наступним кроком буде представлення діаграм послідовності для кожного прецеденту згідно зі сценаріями варіантів використання. Ці діаграми нададуть детальний опис взаємодії між акторами та системою на кожному етапі виконання операцій. Обсяг цих діаграм буде достатнім для виокремлення всіх об'єктів (компонентів) проектованої системи, що дозволить отримати повний огляд функціональності системи та взаємодії її складових частин.

На рисунку 2.7. представлена діаграма послідовності застосування фільтрів в контексті нашої інформаційної системи обліку будівельних інструментів. Ця діаграма надає детальний опис взаємодії між акторами та системою під час виконання операцій із застосуванням фільтрів. У рамках цієї діаграми зображено послідовність дій, які виконуються під час використання фільтрів для пошуку конкретних інструментів. Починаючи з запиту користувача на застосування певних фільтрів, система отримує цей запит і виконує необхідні операції для фільтрації інформації. Після цього система повертає відфільтрований результат користувачеві, який може переглядати відповідні дані відповідно до обраних параметрів фільтрації. Ця діаграма дозволяє отримати чітке уявлення про те, як саме відбувається взаємодія між

користувачами та системою при застосуванні фільтрів, що сприяє розумінню функціональності системи та оптимізації процесу пошуку необхідної інформації.

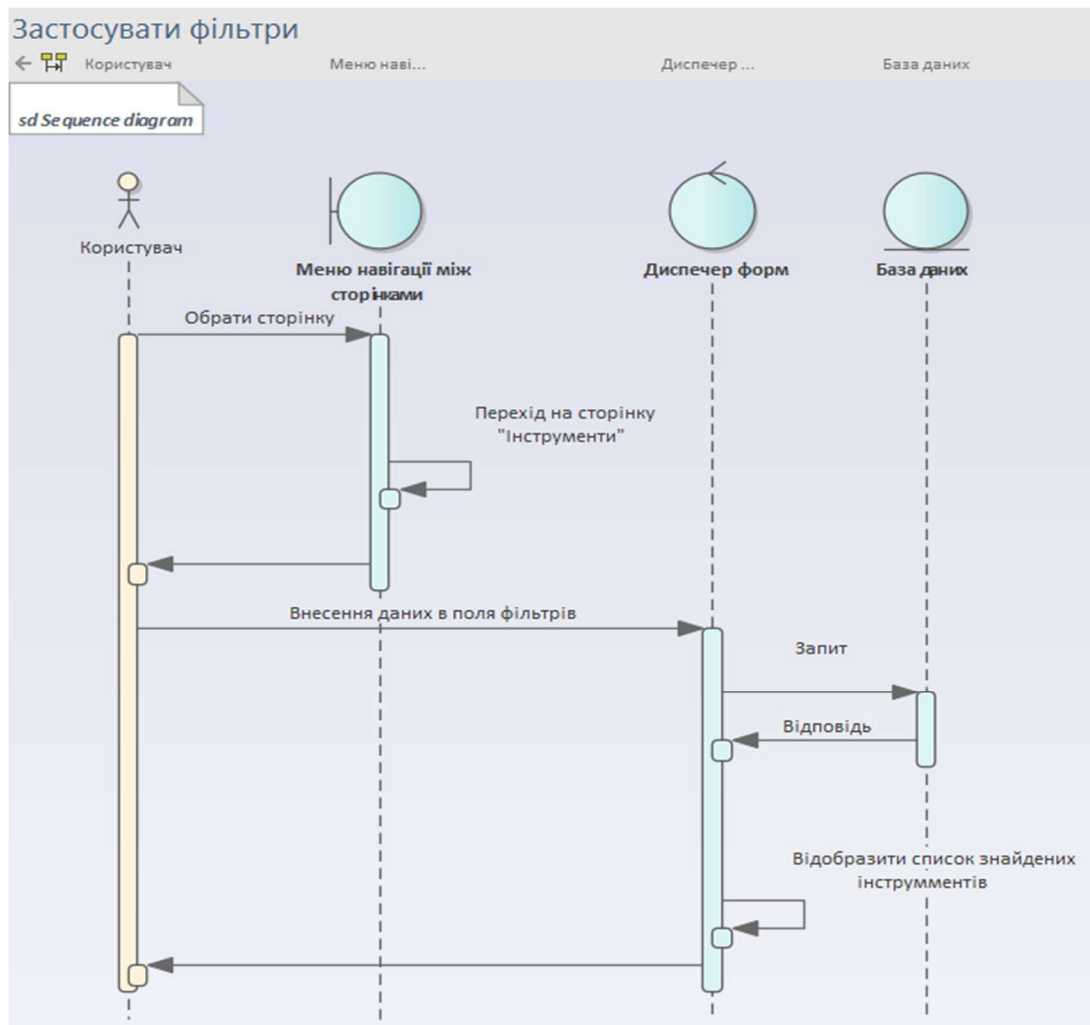


Рисунок 2.7. Діаграма послідовності застосування фільтрів

Діаграма послідовності авторизації ілюструє взаємодію між користувачем і системою під час процедури авторизації. Цей процес включає в себе низку кроків, починаючи від введення користувачем своїх облікових даних (логін та пароль) на відповідній сторінці входу. Після цього система перевіряє введені дані на відповідність збереженим в її базі даних. Якщо вони вірні, користувач успішно авторизується і має доступ до функціоналу системи, в іншому випадку система відображає повідомлення про невірність введених даних і просить користувача повторити процедуру входу. На рисунку 2.8. зображена діаграма послідовності авторизації.

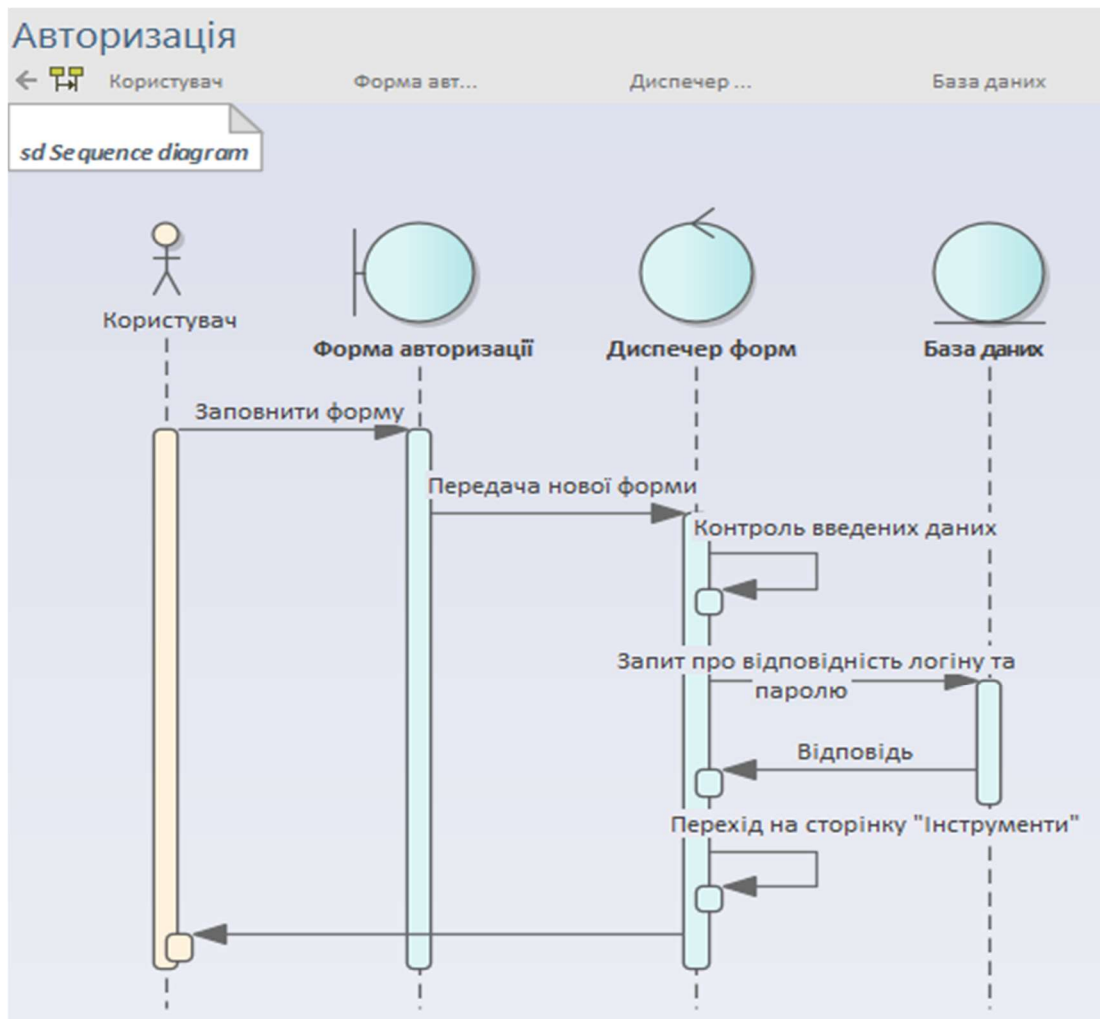


Рисунок 2.8. Діаграма послідовності авторизації

Діаграма послідовності виконання пошуку ілюструє взаємодію між користувачем та системою під час пошуку конкретної інформації. Цей процес включає в себе кілька етапів, починаючи з введення користувачем пошукового запиту на відповідній сторінці. Після цього система отримує запит та проводить пошук в базі даних відповідно до введених критеріїв. Знайдені результати відображаються користувачеві, який може переглядати та обирати потрібну інформацію. На Рисунку 2.9. представлена діаграма послідовності виконання пошуку.

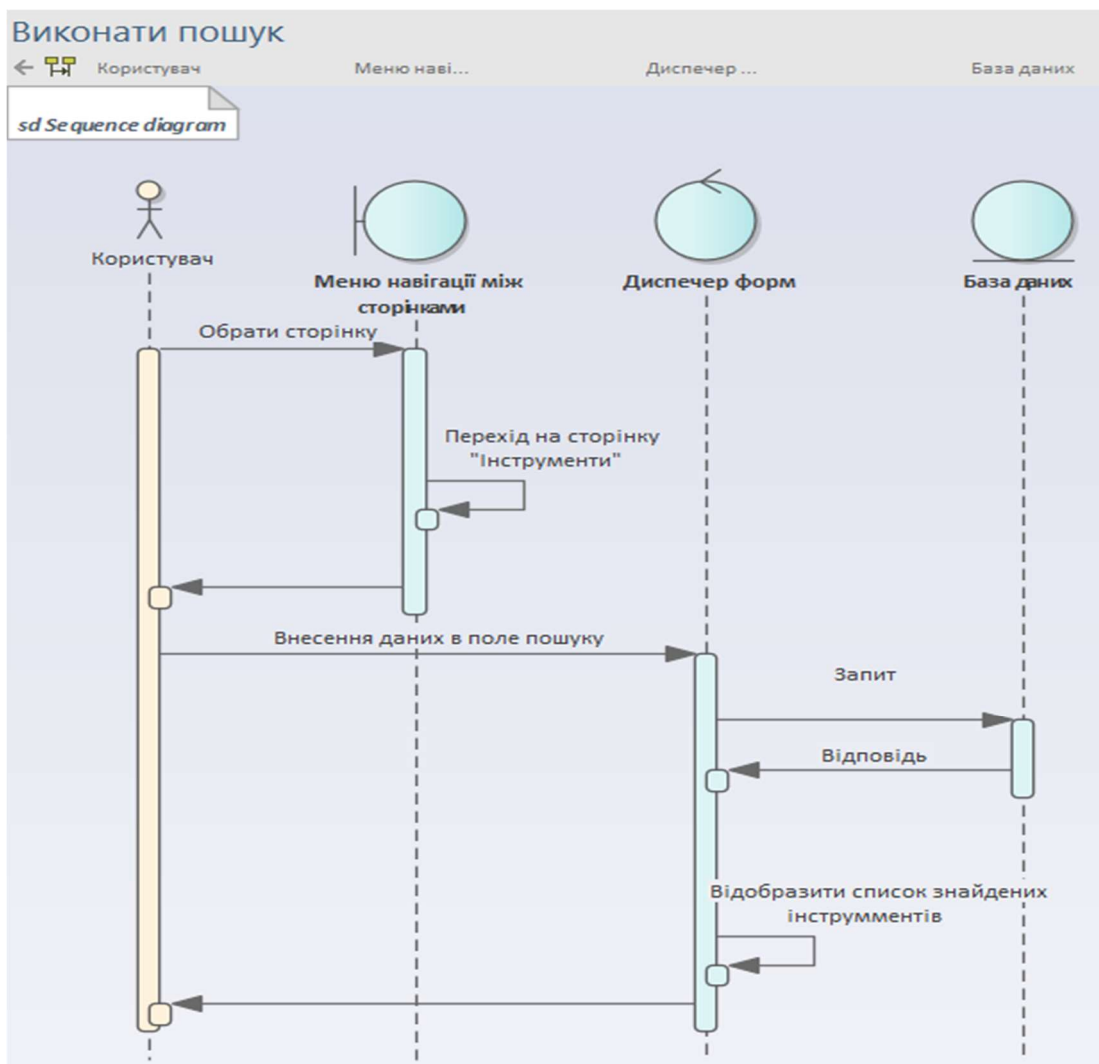


Рисунок 2.9. Діаграма послідовності виконання пошуку

На Рисунок 2.10. зображена діаграма послідовності реєстрації, яка відображає взаємодію між користувачем та системою під час процедури реєстрації в системі. Цей процес включає в себе кілька етапів, починаючи з введення користувачем своїх особистих даних на відповідній сторінці реєстрації. Після цього система перевіряє введені дані на відповідність вимогам та унікальність у системі. Якщо дані є коректними, користувач успішно реєструється в системі і отримує доступ до функціоналу.

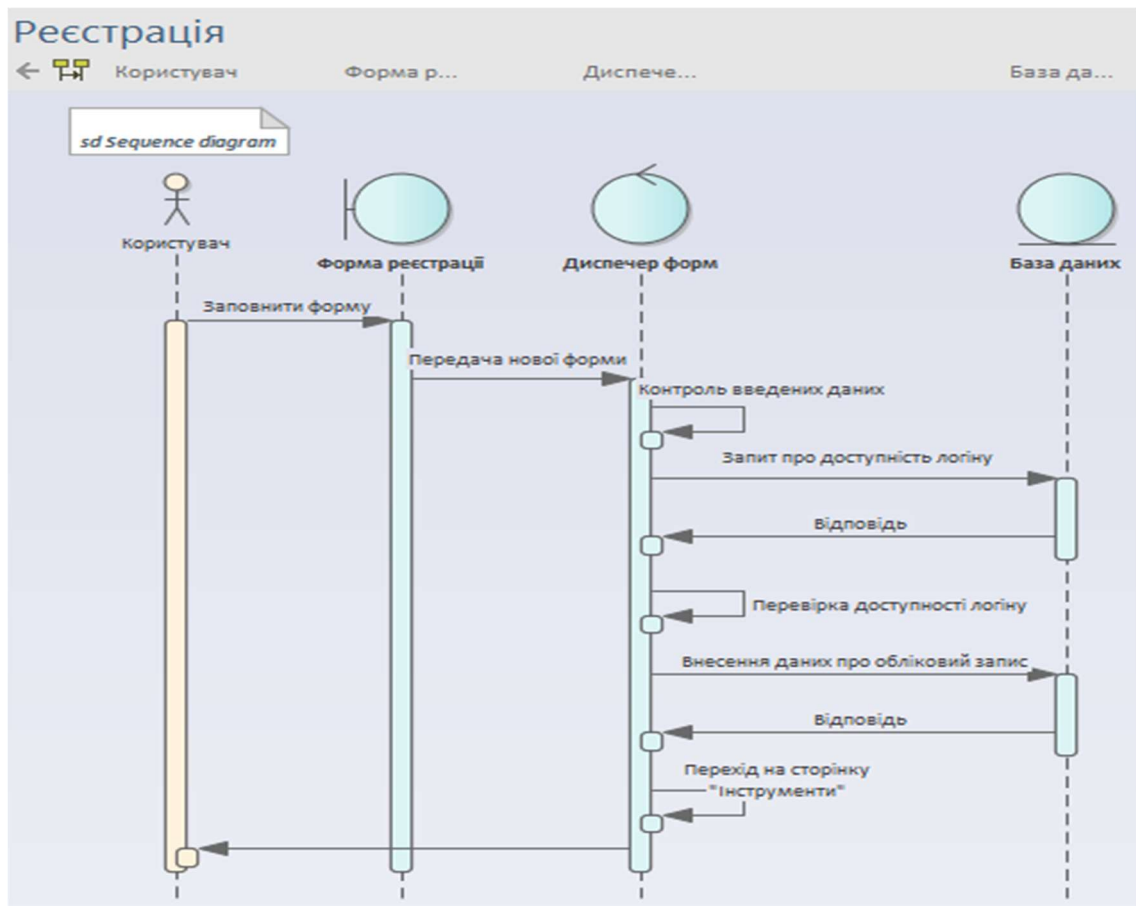


Рисунок 2.10. Діаграма послідовності реєстрації

На Рисунку 2.11. представлена Діаграма послідовності додавання коментарів, яка демонструє взаємодію між користувачем і системою під час процесу додавання коментарів до інструментів або об'єктів. Першим етапом є вибір користувачем конкретного інструмента, до якого він бажає залишити коментар. Далі, користувач вводить текст свого коментаря у відповідне поле на веб-сторінці. Система отримує цей коментар та перевіряє його на відповідність правилам та обмеженням щодо довжини чи формату. Якщо коментар відповідає усім вимогам, він зберігається у базі даних разом із відповідним інструментом.

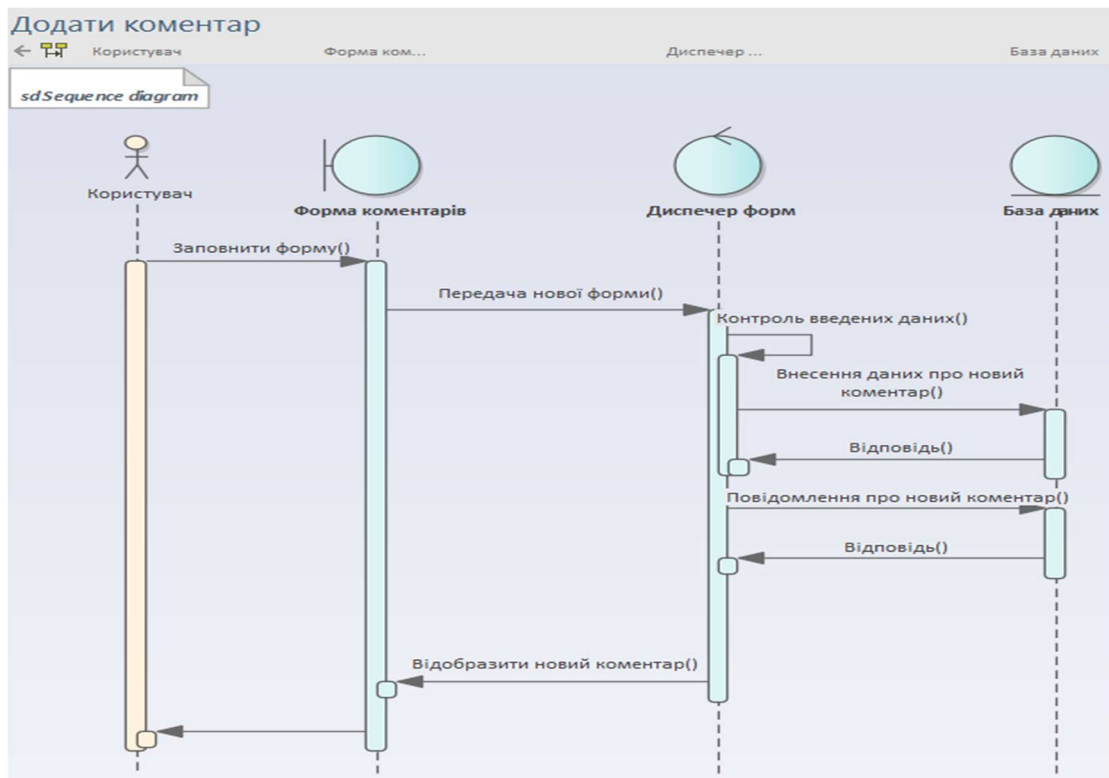


Рисунок 2.11. Діаграма послідовності додавання коментарів

На Рисунку 2.12. зображена діаграма послідовності додавання інструментів, яка відображає взаємодію між користувачем системи та самою системою під час процедури додавання нового інструменту. Користувач вводить дані про новий інструмент, такі як назва, опис, характеристики та інші необхідні дані. Після цього система перевіряє введені дані на відповідність вимогам та правилам заповнення форми. Якщо дані введені коректно, система додає новий інструмент до своєї бази даних та зберігає відповідну інформацію.

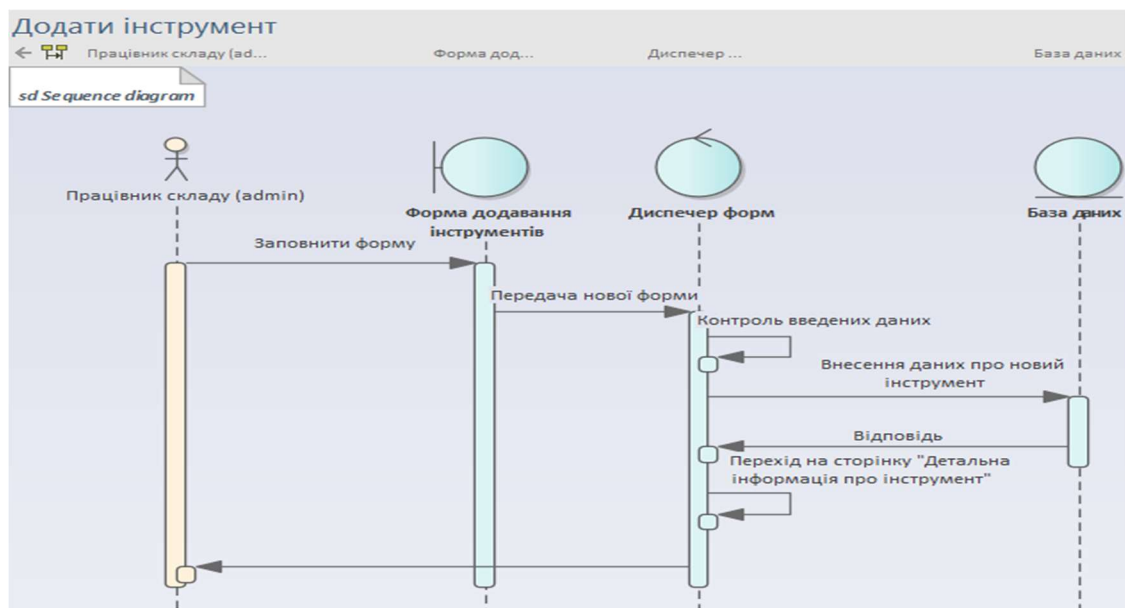


Рисунок 2.12. Діаграма послідовності додавання інструментів

2.3.2. Розподіл вимог за компонентами системи

Діаграму трасування вимог наведено на рисунку 2.13.

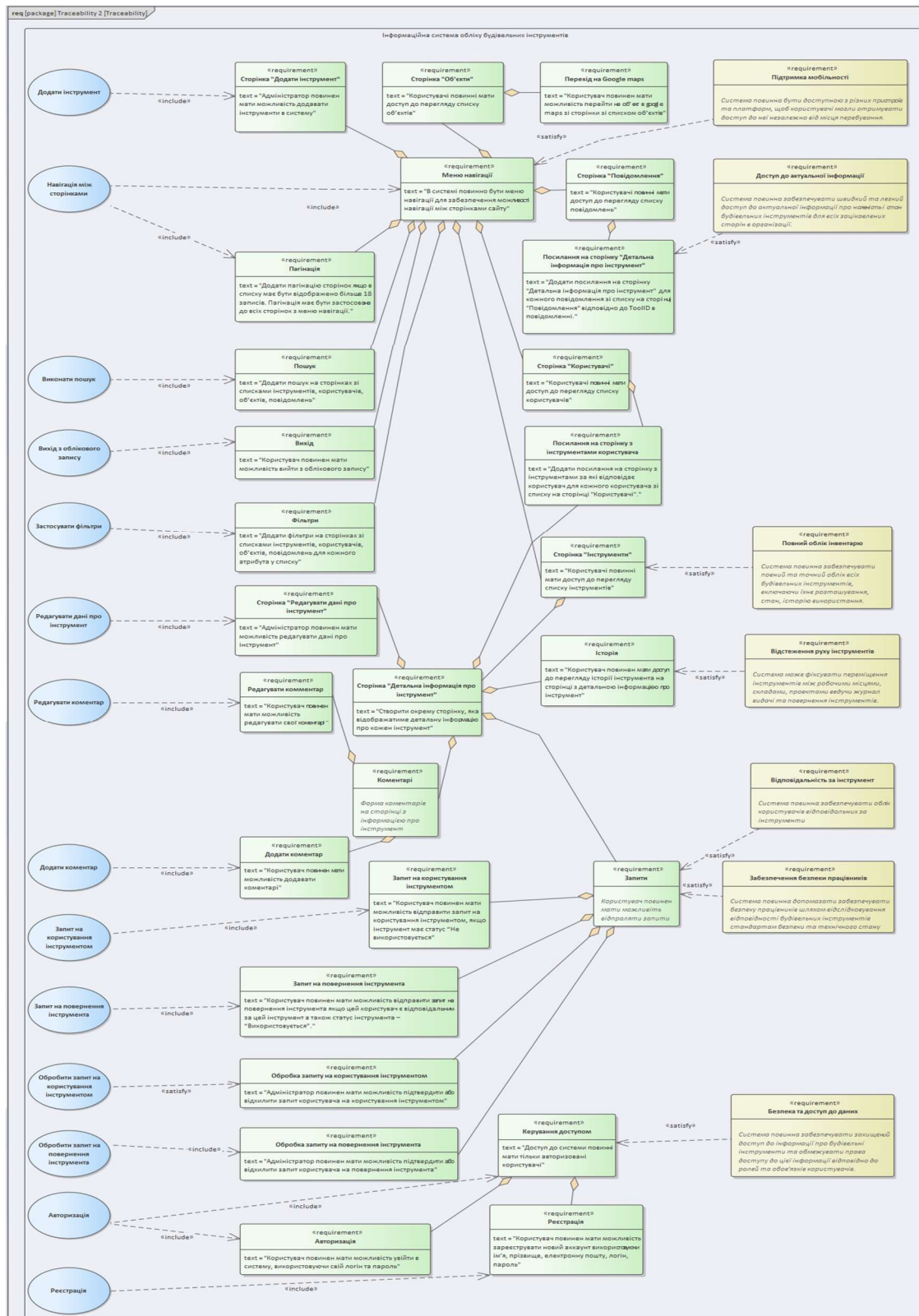


Рисунок 2.13. Діаграма трасування вимог.

Модель структури системи представлена у формі діаграм класів UML, які ілюструють статичну структуру класів системи та зв'язки між ними. Кожен клас на діаграмі представляє окремий компонент системи, а зв'язки між

класами показують взаємозв'язки та залежності між ними. Ці діаграми допомагають зрозуміти організацію та взаємозв'язки між компонентами системи на рівні класів, що сприяє розумінню структури програми та полегшує розробку, тестування та підтримку програмного забезпечення.

Діаграма визначення блоків відображає структуру системи в термінах окремих функціональних блоків або компонентів. Кожен блок на цій діаграмі представляє собою окрему частину системи зі своїми власними функціями та можливостями. За допомогою цієї діаграми можна краще зрозуміти, як система організована на рівні функціональних модулів для досягнення загальних цілей системи. Діаграма визначення блоків є важливим інструментом для архітектурного проектування системи, оскільки дозволяє здійснити аналіз та організацію функціональних компонентів системи перед реалізацією програмного коду. На Рисунку 2.14. представлена діаграма визначення блоків.

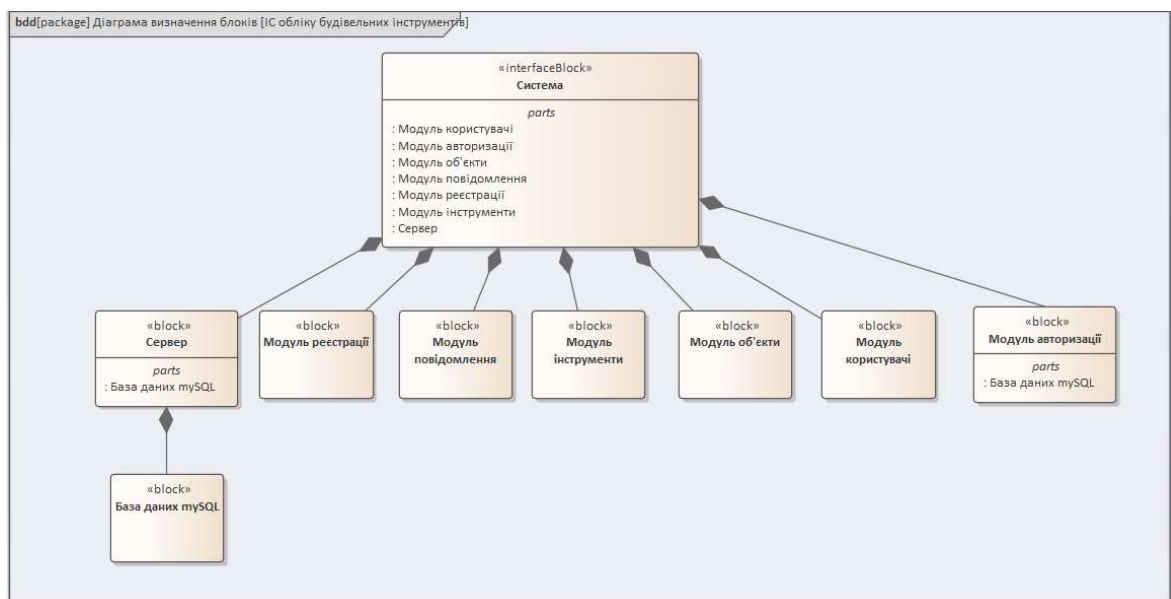


Рисунок 2.14. – Діаграма визначення блоків

На діаграмі внутрішніх блоків представлені внутрішні компоненти або модулі системи, а також їхні взаємозв'язки. У цій діаграмі показані порти - це точки входу або виходу для обміну даними між різними компонентами системи. Порти визначають, як інформація потрапляє в компоненти системи або виходить з них. Вони можуть бути використані для передачі сигналів, даних або подій між різними частинами системи та регулювання їх взаємодії.

Таким чином, на діаграмі внутрішніх блоків порти відображаються як інтерфейси, що забезпечують взаємодію між компонентами системи.

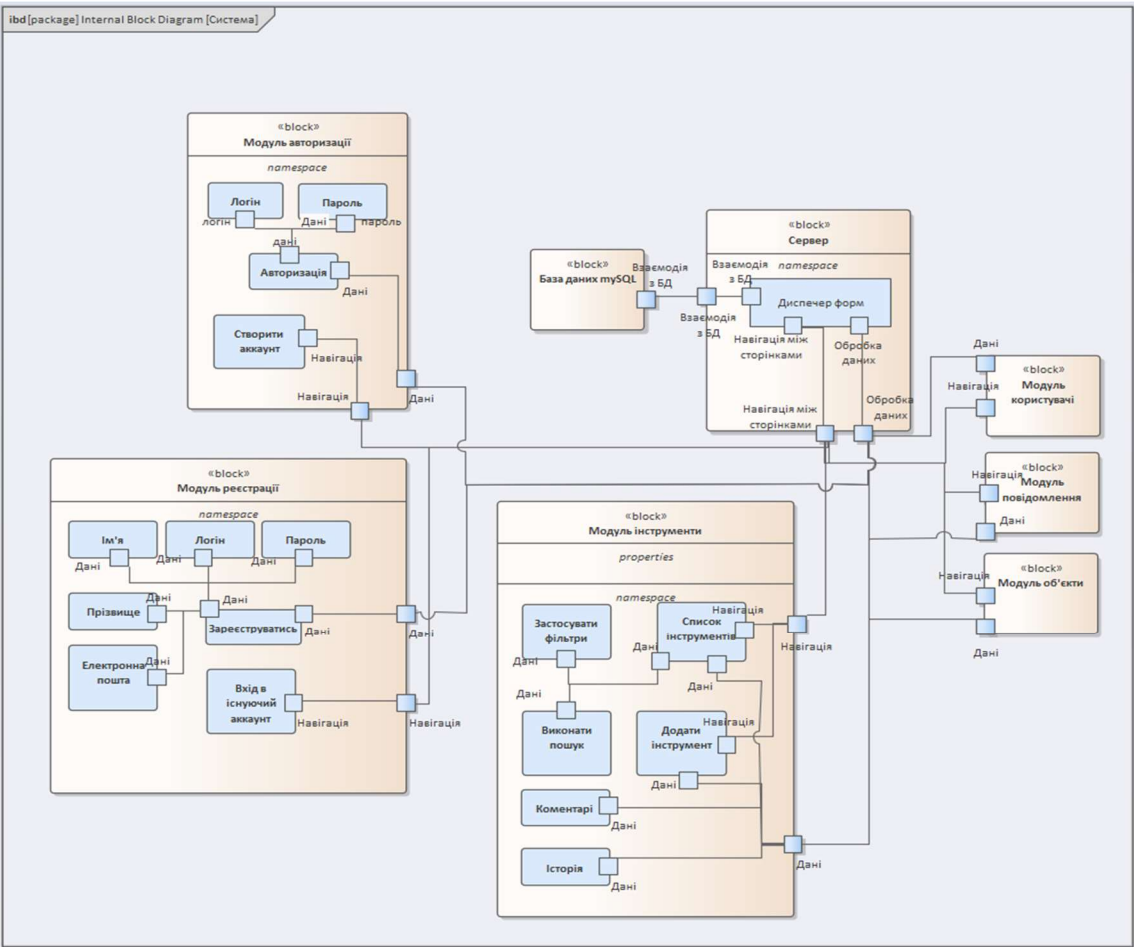


Рисунок 2.15. – Діаграма внутрішніх блоків

Діаграма класів показана на рисунку 2.16.

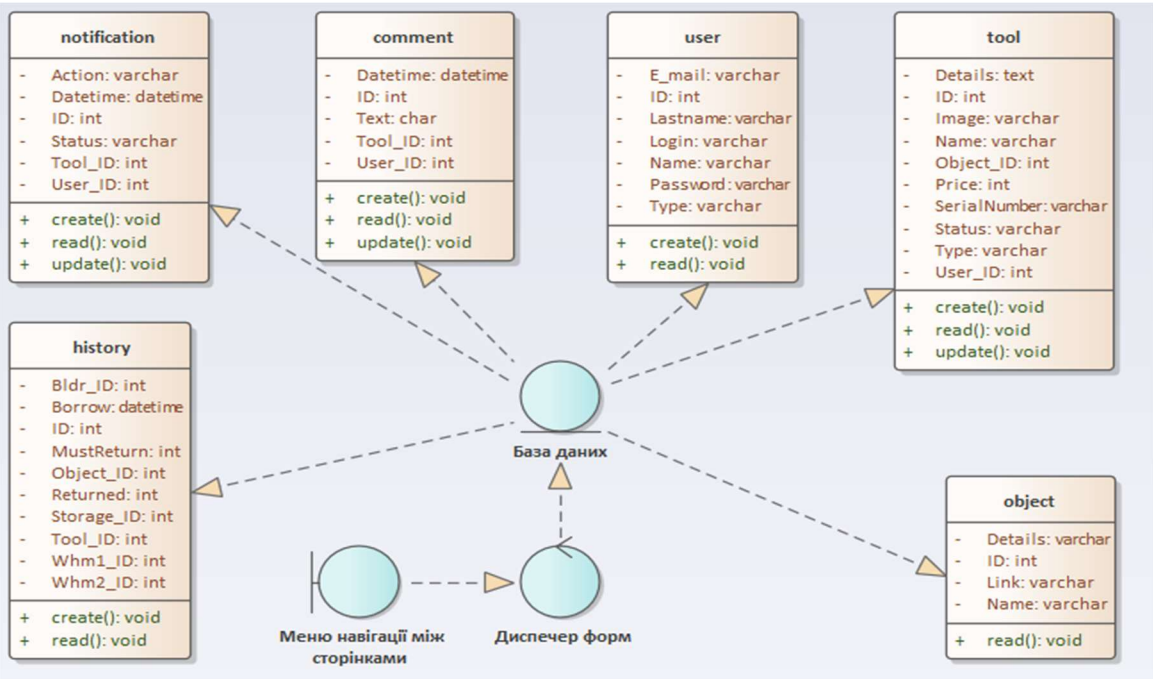


Рисунок 2.16. – Діаграма класів

Матрицю зав’язків прецедентів і функціональних вимог показано на рисунку 2.17.

Рисунок 2.17. – Матриця зав’язків між вимогами та варіантами використання

Матриця зав’язків між функціональними та бізнес вимогами показана на рисунку 2.18.

Target	1	2	3	4	5	6
Source	Запити	Керування доступом	Меню навігації	Посилання на сторі...	Сторінка "Інструме...	Історія
1 Безпека та доступ ...		Satisfy				
2 Відповідальність за...	Satisfy					
3 Відстеження руху і...						Satisfy
4 Доступ до актуально...				Satisfy		
5 Забезпечення безп...	Satisfy					
6 Повний облік інвен...					Satisfy	
7 Підтримка мобільн...			Satisfy			

Рисунок 2.18. – Матриця зав’язків між функціональними та бізнес вимогами

РОЗДІЛ 3. ПРОЕКТУВАННЯ ТА РЕАЛІЗАЦІЯ КОМПОНЕНТІВ СИСТЕМИ

3.1. Інформаційне забезпечення

3.1.1. Загальна характеристика інформаційного забезпечення

Спроектована інформаційна система базується на сервері з веб-інтерфейсом, володіє низкою переваг, що роблять її важливим інструментом для різних типів користувачів. Перш за все, такий підхід забезпечує доступність системи з будь-якого місця та будь-якого пристрою, що має доступ до Інтернету. Це означає, що користувачі можуть працювати з системою навіть за межами офісу, використовуючи для цього свої персональні пристрої, такі як ноутбуки, смартфони або планшети.

Завдяки веб-інтерфейсу, інформаційна система стає більш гнучкою та зручною для використання, оскільки користувачам не потрібно встановлювати жодне спеціалізоване програмне забезпечення на свої пристрої. Достатньо лише відкрити веб-браузер і ввести адресу системи для отримання доступу до всіх її функцій та можливостей.

Такий підхід також полегшує процес супроводження та оновлення системи, оскільки всі необхідні зміни можуть бути внесені на серверному боці безпосередньо адміністратором системи. Користувачам не потрібно втручатися у процес оновлення або встановлення патчів, оскільки всі зміни відбуваються централізовано на сервері.

Крім того, використання серверної бази даних забезпечує надійне зберігання та обробку інформації, а також забезпечує її безпеку та цілісність. За допомогою веб-інтерфейсу можливе створення різних рівнів доступу до інформації для різних типів користувачів, що дозволяє керувати правами доступу та забезпечувати конфіденційність даних.

3.1.2. Організація збору і передавання первинної інформації

Перше повідомлення, "Форма реєстрації користувача", що наведено наведено у таблиці 2.2. є інтерактивною екранною формою, яку використовують нові користувачі для реєстрації в системі. Ця інформація передається на сервер через введення даних користувачем.

"Запит на позичення інструменту" та "Запит на повернення інструменту" також є формами на сайті, через які будівельники звертаються з запитом на позичення або повернення інструментів. Ці дані відправляються серверу для обробки.

"Додавання нового інструменту" та "Редагування даних про інструмент" — це опції, які доступні працівникам складу для додавання нових інструментів або внесення змін до існуючих даних про інструменти. Вони також використовують екранні форми для введення інформації, яка надсилається на сервер для збереження в базі даних.

"Форма авторизації користувача" — це форма, яка використовується для входу в систему існуючим користувачам. Дані з цієї форми надсилаються на сервер для перевірки прав доступу.

"Відповідь на запит користування інструментом" та "Відповідь на запит повернення інструмента" — це повідомлення, які генеруються працівниками складу у відповідь на запити від будівельників. Вони також надсилаються на сервер для обробки.

"Додати коментар" — це можливість, яку використовують користувачі для додавання коментарів до інструментів чи інших об'єктів у системі. Коментарі також передаються на сервер через відповідну форму для збереження в базі даних.

Перейдемо до вихідного повідомлення "Інформація про інструменти". Це повідомлення надсилається користувачам після переходу на сторінку зі списком інструментів. Воно містить важливі дані про кожен інструмент, такі як назва, опис, користувач тощо. Ця інформація надає користувачам повний огляд інструментів, доступних у системі, та допомагає їм приймати обґрунтовані рішення щодо їх використання.

"Список коментарів" дозволяє користувачам переглядати коментарі, залишені іншими користувачами щодо конкретного інструменту. Це дає можливість користувачам ділитися враженнями, досвідом та порадами щодо використання інструментів, що сприяє підвищенню спільної взаємодії та обміну знаннями.

"Історія користування інструментом" надає користувачам змогу переглядати історію використання конкретного інструменту. Це дозволяє користувачам відстежувати, хто і коли використовував інструмент, що може бути корисним для планування робіт та вирішення можливих конфліктів у використанні інструментів.

"Список об'єктів" відображає користувачам перелік об'єктів, що доступні для перегляду або використання. Це може бути корисно для будівельників або інших співробітників, які шукають конкретні об'єкти для роботи або огляду.

"Список повідомлень" надає користувачам зручний спосіб отримання сповіщень та повідомлень щодо різних подій або оновлень у системі. Це може включати повідомлення про стан інструментів, рекомендації щодо використання, оброблені чи відправлені запити, або будь-які інші важливі повідомлення.

"Список користувачів" дозволяє адміністраторам або іншим користувачам переглядати інформацію про інших користувачів системи. Це може включати їхні дані контакту, ролі в системі або іншу важливу інформацію, яка допомагає в управлінні користувачами та наданні доступу до ресурсів системи.

Всі ці дані отримуються з бази даних системи та надсилаються користувачам через екранні форми на веб-сайті. Такий підхід забезпечує ефективну і швидку взаємодію користувачів з системою, дозволяючи їм легко отримувати доступ до необхідної інформації та функцій.

3.1.3. Побудова системи класифікації та кодування

У роботі не було використано конкретні методи кодування для класифікації об'єктів. Замість цього, об'єкти були організовані за допомогою інших параметрів або характеристик, таких як тип, призначення чи категорія. Відсутність потреби в кодуванні була обумовлена специфікою системи та особливостями даних, що оброблялися.

Для класифікації інструментів можна використовувати типи, які відображають їх призначення або категорії. При додаванні інструменту

працівник складу визначає тип і вказує його у відповідному полі. Приклади типів інструментів:

Електроінструменти:

- Дрилі
- Шліфувальні машини
- Електрофрези
- Електропилки

Ручні інструменти:

- Молотки
- Лещата
- Гайкові верти
- Лобзики

Вимірювальні інструменти:

- Лазерні вимірювачі
- Рулетки
- Терези
- Геодезичні інструменти

Садові інструменти:

- Газонокосарки
- Граблі
- Газонні тримачі
- Садові ножиці

Щодо класифікації повідомлень за властивістю "Дія", це включає такі категорії, як:

- Запит на отримання
- Запит на повернення
- Підтверджено отримання
- Відхилено отримання
- Підтверджено повернення
- Відхилено повернення

- Новий коментар

Також повідомлення поділяються на "прочитані" і "не прочитані", що відображає статус переглянутих або непрочитаних повідомлень користувачем.

3.1.4. Проектування форм первинних документів

Розробивши структуру інтерфейсів у вигляді ієрархії екранів, ми надаємо користувачам зручну та логічну навігацію по системі. На рисунку 3.1. представлено результат цієї роботи. Починаючи свою роботу з системою, користувач спочатку переходить на екран реєстрації, де може створити обліковий запис або перейти до входу в існуючий запис. Після успішної авторизації користувач потрапляє на головний екран, де він може бачити список інструментів. З цього екрану він має можливість перейти на інші сторінки системи в залежності від своїх потреб. Наприклад, сторінка "Об'єкти" відображає інформацію про склади та об'єкти будівництва, а сторінка "Користувачі" містить список всіх користувачів системи. Функціонал сторінки "Додати інструмент" відповідає її назві, дозволяючи користувачеві додавати нові інструменти в систему. Сторінка "Детальніше про інструмент" надає повну інформацію про кожен інструмент, що знаходиться в системі. Крім того, на цій сторінці доступні форми коментарів та історія інструмента. Сторінка "Редагувати дані про інструмент" надає користувачеві можливість редагувати дані про інструмент, якщо він має відповідні права адміністратора. Це забезпечує відповідну безпеку та контроль інформації у системі.

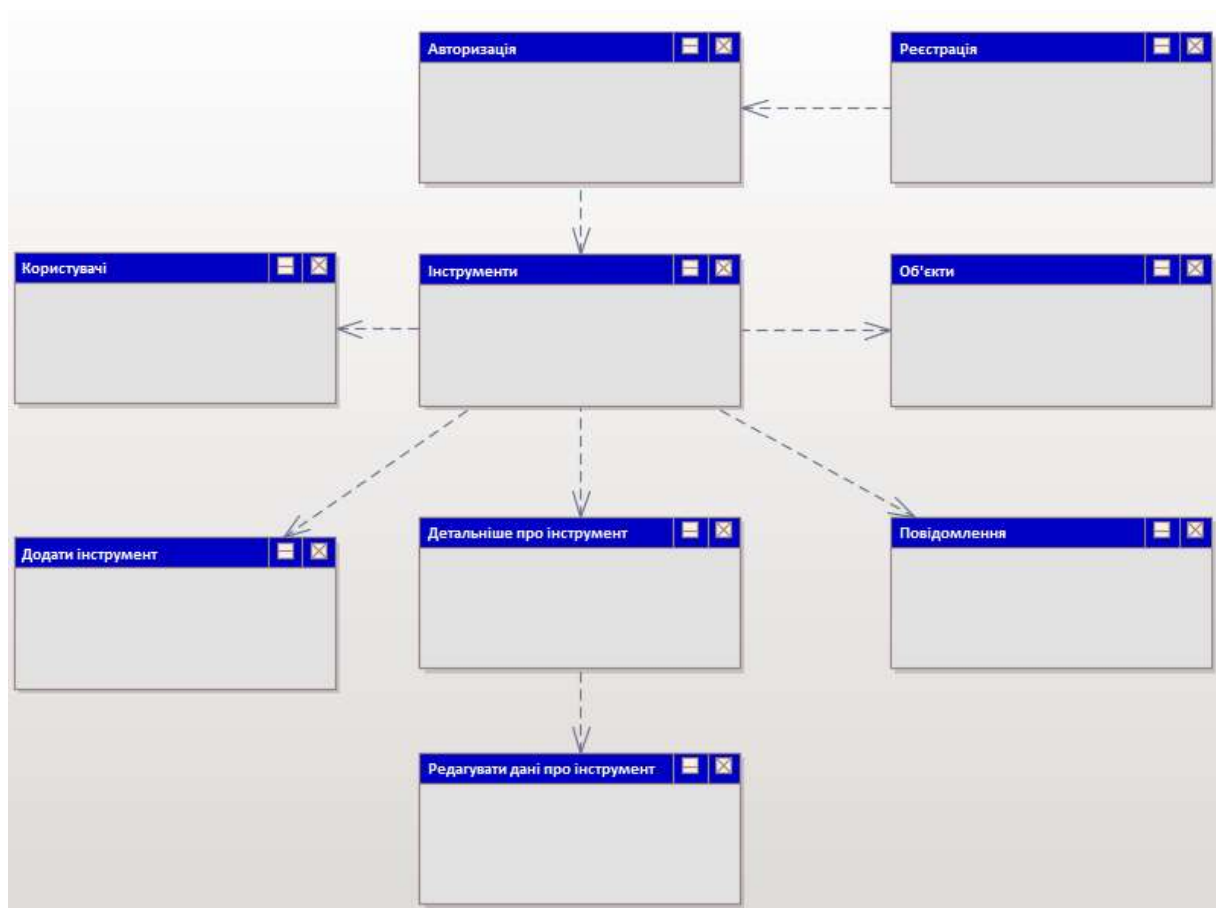


Рисунок 3.1. Структура інтерфейсу у вигляді ієрархії екранів

На наступних рисунках представлені діаграми інтерфейсу, які відображають кілька екранних форм, побудованих у програмі Enterprise Architect. Ці діаграми демонструють структуру та елементи кожної форми, а також приблизне розташування цих елементів на сторінці. Кожен елемент інтерфейсу, такий як кнопки, текстові поля, списки тощо, ретельно розміщений на кожній формі з урахуванням їхньої функціональності та зручності для користувача.

У діаграмах можна побачити взаємодію між різними елементами інтерфейсу, наприклад, як кнопки або посилання можуть викликати певні дії або переходи до інших екранів. Крім того, вони відображають зовнішній вигляд кожної форми, включаючи розміщення тексту, зображень та інших графічних елементів.

На сторінці авторизації користувач має ввести логін та пароль від існуючого аккаунта для того щоб авторизуватись, або можна перейти до створення нового аккаунта.

Рисунок 3.2. Екранна форма “Авторизація”

Сторінка “Інструменти” є одною з найважливіших, адже тут користувачі можуть знайти основну інформацію про інструменти.

Рисунок 3.3. Екранна форма “Інструменти”

На сторінці “Детальніше про інструмент” окрім отримання детальної інформації, можна також відправляти або обробляти запити, а також додавати коментарі.

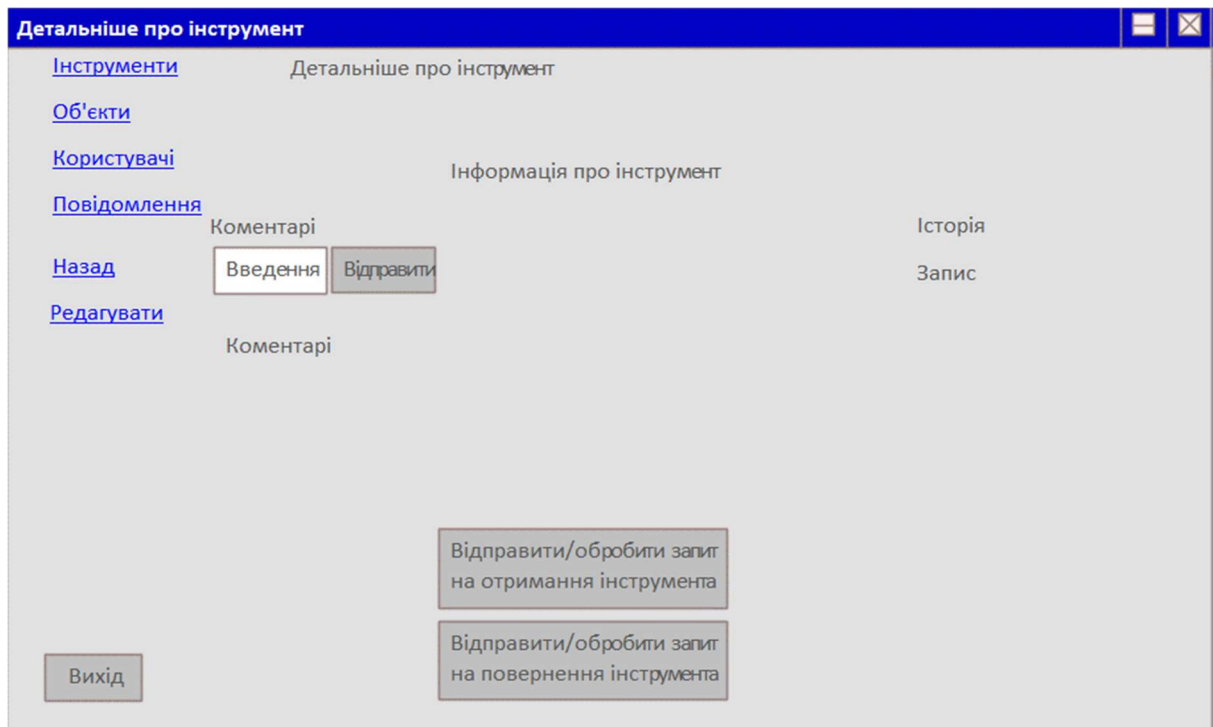


Рисунок 3.4. Екранна форма “Детальніше про інструмент”

3.1.5. Вибір СКБД

MySQL— вільна реляційна система керування базами даних (СКБД), яка працює як сервер для забезпечення багатокористувацького доступу до великої кількості баз даних.

Проекти на основі безкоштовного ПЗ, які вимагають повнофункціональної системи керування базами даних часто використовують MySQL. До таких проектів відносяться, наприклад, WordPress, phpBB, Drupal та інше програмне забезпечення, побудоване на стеку продуктів LAMP (Linux, Apache, MySQL, PHP/Perl/Python). MySQL також використовується в багатьох гучних великомасштабних Web-продуктах, включаючи Wikipedia, Google (для програми AdWords), Facebook, YouTube, Flickr, Yahoo!, Digg, LiveJournal, Nokia тощо.

MySQL— компактний багато потоковий (багатонитевий) сервер баз даних. Характеризується великою швидкістю, стійкістю і простотою використання.

Для некомерційного використання MySQL є безкоштовним. Можливості сервера MySQL:

- простота у встановленні та використанні;

- підтримується необмежена кількість користувачів, що одночасно працюють із БД;
- кількість рядків у таблицях може досягати 50 млн.;
- висока швидкість виконання команд;
- наявність простої і ефективної системи безпеки. [1]

Отже при виборі системи керування базами даних для нашого проекту, однією з ключових розглянутих альтернатив була MySQL. Рішення вибрати саме цю СКБД було обґрунтоване кількома факторами, які виявилися критичними для успішної реалізації проекту.

По-перше, MySQL відома своєю широкою популярністю та розповсюдженістю. Це важливо, оскільки це означає, що існує велика спільнота користувачів і розробників, які надають підтримку, розвивають різноманітні додатки та допомагають у вирішенні можливих проблем. Така активна спільнота створює великий обсяг документації, яка є дуже корисною під час розробки і підтримки проекту.

По-друге, MySQL є відкритою системою з вільним доступом. Це означає, що ми можемо використовувати її безкоштовно, а також мати доступ до вихідного коду. Такий підхід дає нам більшу гнучкість та контроль над тим, як ми можемо налаштовувати і пристосовувати базу даних до наших потреб.

Крім того, MySQL відома своєю швидкодією та ефективністю. Вона може ефективно обробляти великі обсяги даних та виконувати запити швидко, що дозволяє підтримувати високу продуктивність системи навіть при інтенсивному навантаженні.

Ще однією важливою перевагою MySQL є її добра сумісність з багатьма іншими технологіями та мовами програмування. Це дозволяє нам інтегрувати її з існуючими або майбутніми складовими нашого проекту без особливих труднощів.

Отже, обираючи MySQL, ми врахували не лише його технічні характеристики, а й широке сприйняття в галузі, відкритий характер, ефективність та сумісність з іншими технологіями, що робить її ідеальним вибором для нашого проекту.

3.1.6. Інфологічна модель бази (сховища) даних

Мета інфологічного моделювання – забезпечення найбільш природних для людини способів збору та представлення тієї інформації, яку передбачається зберігати в базі даних. Тому інфологічну модель даних намагаються будувати по аналогії з природною мовою.

На етапі даталогічного проектування будується логічна структура БД. При цьому даталогічна модель розробляється з урахуванням конкретної реалізації СУБД основі її інфологічної моделі. Кінцевим результатом даталогічного проектування є опис структури БД на мові опису даних конкретних СУБД.

Фізична модель – прив'язка даталогічної моделі БД до середовища зберігання. Використовуються можливості даної конкретної СУБД. Приховано від розробника.

Основними конструктивними елементами інфологічної моделі є сутність, зв'язки між ними та їх властивості (атрибути)[2].

Сутність - це не лише окремий об'єкт, але й основний будівельний блок у будь-якій базі даних. Вона відображає реальний або уявний об'єкт, про який потрібно зберігати інформацію. Завдяки сутностям ми можемо структурувати дані в базі даних, визначаючи основні області інформації, які потрібно зберігати та обробляти. Наприклад, якщо ми маємо базу даних про компанію, то сутностями можуть бути "співробітник", "відділ", "продукт" тощо.

Атрибут - це конкретна характеристика сутності, яка дозволяє нам описати або ідентифікувати її. Це можуть бути різноманітні властивості або параметри, що характеризують об'єкт. Наприклад, для сутності "користувач" атрибутами можуть бути "ім'я", "прізвище", "адреса", "електронна пошта" тощо. Атрибути визначаються для кожної сутності окремо і мають свої типи даних.

Ключ - це особливий атрибут або група атрибутів, які однозначно ідентифікують кожний екземпляр сутності. Ключі використовуються для унікальної ідентифікації записів в таблицях бази даних і забезпечення їх

правильної адресації. Вони грають важливу роль у забезпеченні цілісності та ефективного доступу до даних.

Зв'язок між сутностями в базі даних відображає їх взаємозв'язок та визначає, як вони пов'язані між собою. Без зв'язків база даних була б простою, але непрактичною, оскільки користувачі не могли б здійснювати ефективний пошук та отримання потрібної інформації. Зв'язки дозволяють забезпечити зручний доступ до даних, встановлюючи логічні зв'язки між різними сутностями.

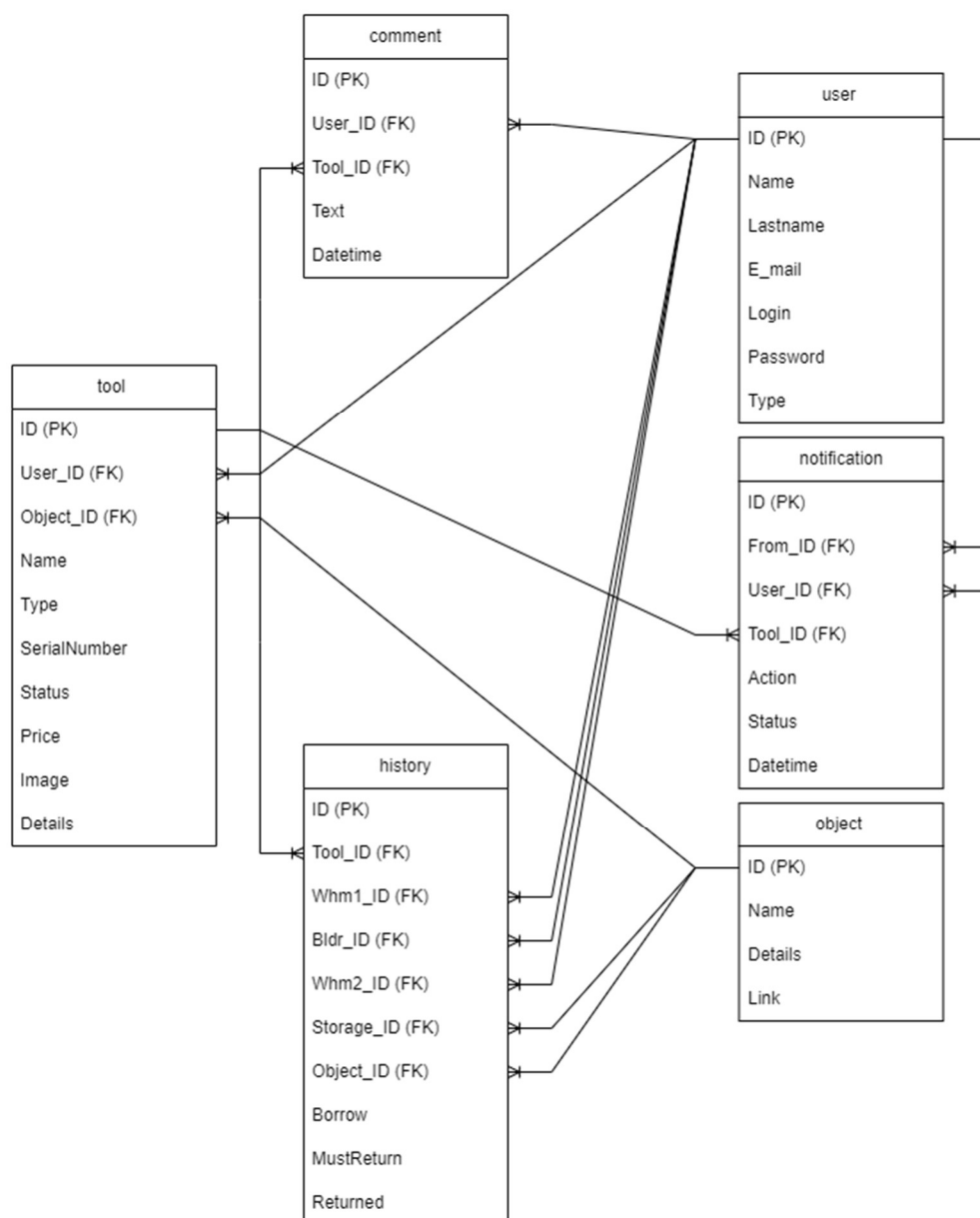


Рисунок 3.5. Інфологічна модель БД

3.1.7. Даталогічна модель бази (сховища) даних

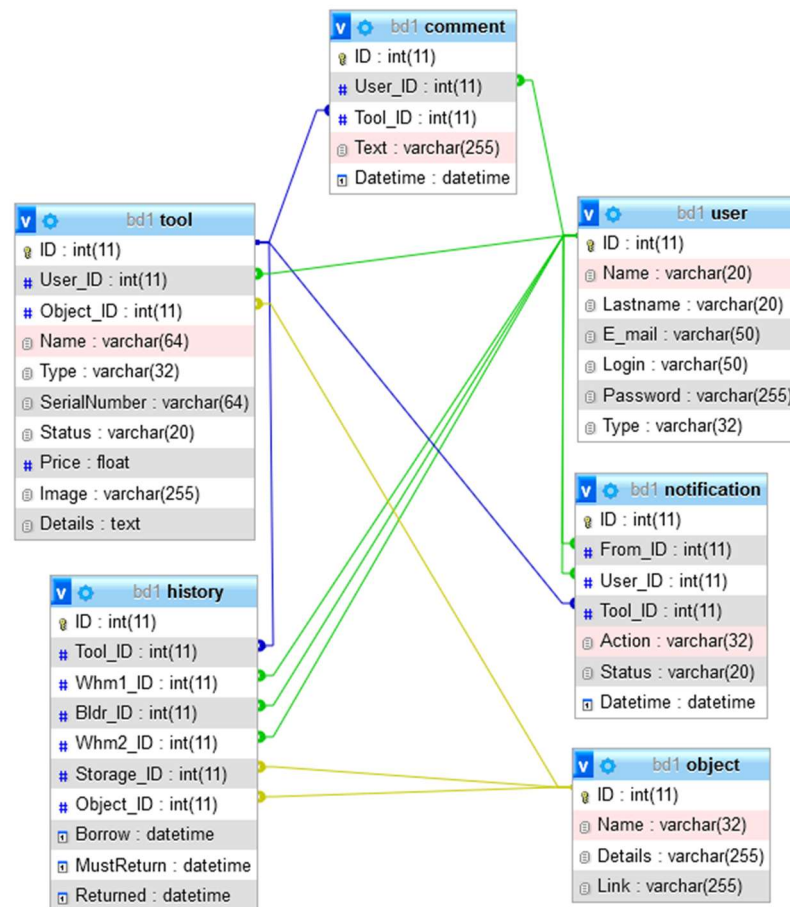


Рисунок 3.6. Модель БД в середовищі СКБД

Наступний етап проектування системи включає в себе доповнення моделі баз даних і додатків діаграмами їх розміщення на технічних засобах.

На даному етапі розглядаються такі поняття UML, як:

- компонент – елемент фізичного представлення системи, який реалізує певний набір інтерфейсів;
- залежність – зв’язок між двома елементами, що позначає ситуацію, при якій зміна одного елемента моделі тягне за собою зміну її іншого елемента;
- процесор і пристрій – вузол, що виконує і не виконує обробку даних відповідно;
- з’єднання – зв’язок між процесорами і пристроями.

Найбільш повно особливості фізичного представлення системи на мові UML дозволяють реалізувати діаграми компонентів і діаграми розгортання[3].

Діаграма компонентів — це важливий інструмент у проектуванні програмного забезпечення, який допомагає візуалізувати та організувати компоненти системи та їх зв'язки. Вона використовується для моделювання архітектури системи на високому рівні абстракції, дозволяючи розуміти взаємозв'язки між компонентами, їх функціональність та взаємодію з іншими частинами системи.

На діаграмі компонентів зазвичай зображуються компоненти системи у вигляді прямокутників або блоків, а зв'язки між компонентами — у вигляді стрілок або ліній. Кожен компонент може мати властивості, методи та інші характеристики, які можуть бути відображені на діаграмі.

Ця діаграма допомагає команді розробників та архітекторам системи краще зрозуміти структуру проекту, його модульність та логіку взаємодії між компонентами. Вона також може бути використана для управління залежностями між компонентами та визначення стратегій розгортання системи.

На діаграмі компонентів відображають такі елементи:

- Компоненти: Основні будівельні блоки системи, такі як модулі, бібліотеки, сервіси тощо.
- Інтерфейси: Способи взаємодії між компонентами, які визначаються зовнішніми та внутрішніми API.
- Залежності: Взаємозв'язки між компонентами, які вказують на те, як один компонент використовує інший.
- Ресурси: Використані ресурси, такі як бази даних, сервери, зовнішні API тощо.

Діаграма компонентів є важливою частиною процесу розробки програмного забезпечення, оскільки вона допомагає встановити загальну структуру системи та спростити розуміння її складної архітектури. Діаграму компонентів зображено на рисунку 3.7.

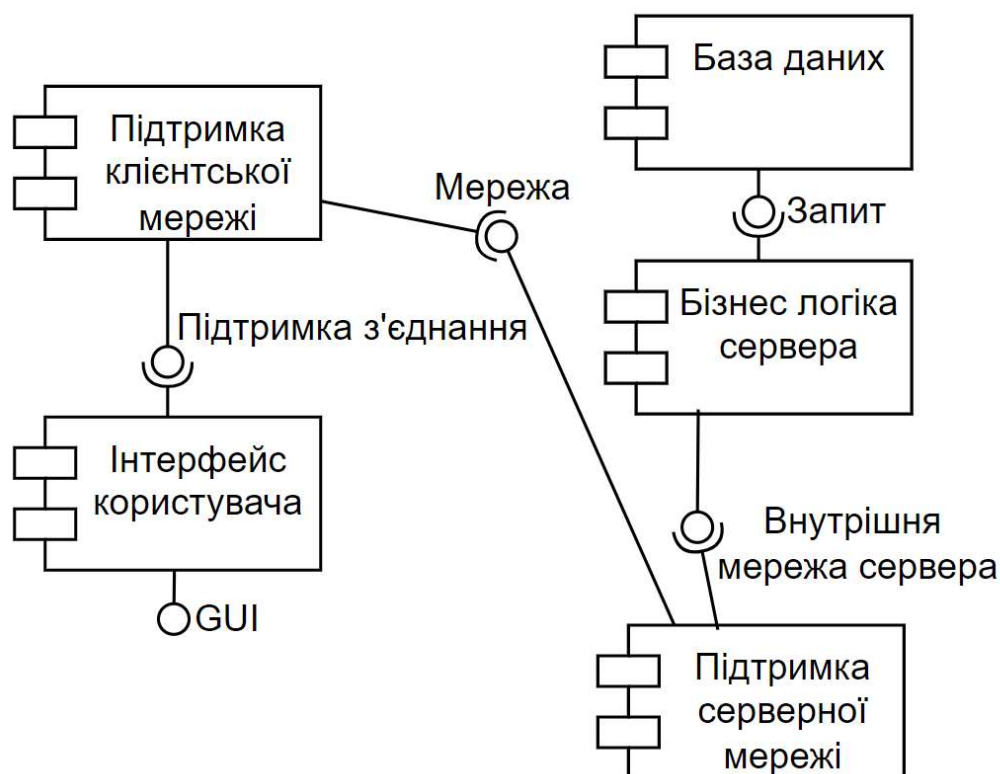


Рисунок 3.7. Діаграма компонентів

Діаграма розгортання — це засіб візуалізації та моделювання архітектури системи, який дозволяє показати фізичне розташування компонентів програмного забезпечення та їх зв'язки з апаратними пристроями або середовищем виконання. Ця діаграма надає інформацію про розгортання компонентів системи на реальних апаратних пристроях або серверах, що допомагає зрозуміти, як саме система буде працювати в реальному середовищі.

Діаграми розгортання використовуються з єдиною метою опису того, як програмне забезпечення розгортається в апаратну систему. Він візуалізує, як програмне забезпечення взаємодіє з апаратним забезпеченням для виконання всіх функцій. Він використовується для опису взаємодії програмного забезпечення з обладнанням і навпаки[4].

У діаграмі розгортання компоненти програмного забезпечення відображаються у вигляді прямокутників або кругів, а апаратні пристрої — у вигляді прямокутників чи овалів. Зв'язки між компонентами та апаратними пристроями показуються стрілками або лініями, що вказують на напрямок комунікації.

Основні елементи діаграми розгортання:

- Компоненти програмного забезпечення: Приклади цих компонентів можуть включати веб-сервери, бази даних, додатки та інші програмні модулі.
- Апаратні пристрої: Це фізичне обладнання, на якому виконується програмне забезпечення, таке як сервери, комп'ютери, маршрутизатори тощо.
- Мережеві зв'язки: Показують зв'язки між апаратними пристроями та компонентами програмного забезпечення через мережу, таку як Інтернет або локальну мережу (LAN).
- Артефакти: Інші елементи, такі як конфігураційні файли, бази даних або інші ресурси, які можуть бути розгорнуті на апаратних пристроях.

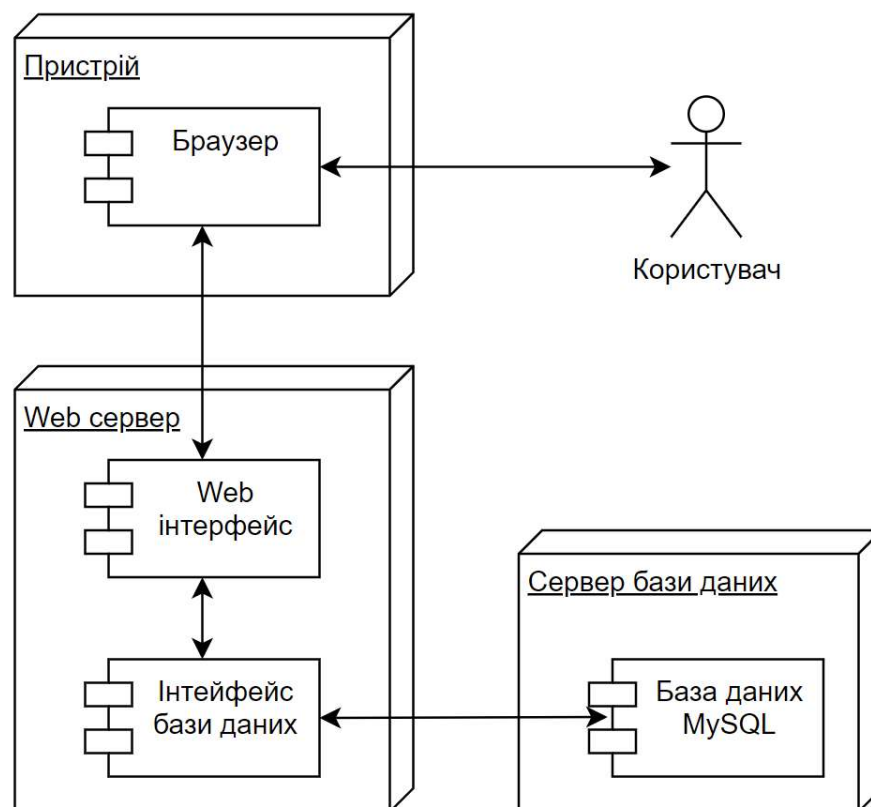


Рисунок 3.8. Діаграма розгортання

В додаток 2 винесено код експортованої БД MySQL.

3.2. Технічне забезпечення

Загальні положення та схема автоматизації

У рамках розробленої системи обліку будівельних інструментів, передбачається наявність серверного обладнання та робочих місць для

користувачів. Система буде розміщена на власному сервері компанії для забезпечення повного контролю над даними та безпекою інформації. Користувачі матимуть можливість доступу до системи через веб-інтерфейс з будь-якого пристрою, що має підключення до Інтернету.

Структура комплексу технічних засобів

Для забезпечення функціонування системи необхідно мати наступне обладнання:

- Сервер: для зберігання бази даних, обробки запитів користувачів та надання сервісів.
- Робочі станції: для користувачів системи, які будуть мати доступ до основного функціоналу через веб-інтерфейс.
- Мережеве обладнання: для підключення робочих станцій до сервера та забезпечення стабільної роботи мережі.

Опис автоматизованого робочого місця (АРМ)

Кожне робоче місце в системі включатиме в себе ноутбук або ПК чи смартфон з встановленим веб-браузером для доступу до системи через веб-інтерфейс. Мінімальні вимоги для персонального комп'ютера (ПК) для коректної роботи системи включають наступне:

- Операційна система: Windows 7 або новіша, macOS 10.12 або новіша, або будь-яка сучасна версія Linux.
- Процесор: Двоядерний процесор з тактовою частотою не менше 2,0 ГГц.
- Оперативна пам'ять (RAM): Мінімум 4 ГБ.
- Внутрішнє сховище: Мінімум 128 ГБ вільного місця на жорсткому диску або SSD для зберігання додатків та даних.
- Дисплей: Роздільна здатність екрану HD (1280x720 пікселів) або вище.
- Графічна карта: Інтегрована або віддільна графічна карта, яка підтримує прискорення графіки OpenGL 3.3 або новіше.

- Мережеві можливості: Підтримка Ethernet або Wi-Fi для з'єднання з мережею Інтернет.
- Браузер: Остання версія Google Chrome, Mozilla Firefox, Safari або Microsoft Edge для використання веб-клієнта системи "BuildMaster".
- Інше обладнання: Клавіатура, миша та монітор для введення та відображення інформації.

Мінімальні вимоги до смартфона для коректної роботи системи включають:

- Операційна система: Android 6.0 або новіша, iOS 10 або новіша.
- Процесор: Чотирьохядерний процесор з тактовою частотою не менше 1,5 ГГц.
- Оперативна пам'ять (RAM): Мінімум 2 ГБ.
- Внутрішня пам'ять: Мінімум 16 ГБ для зберігання додатків та даних.
- Дисплей: Роздільна здатність екрану HD (1280x720 пікселів) або вище, діагональ екрану 4,5 дюйма або більше.
- Бездротові технології: Підтримка Wi-Fi для з'єднання з мережею Інтернет та Bluetooth для обміну даними.
- Батарея: Мінімум 2500 мАг для забезпечення тривалої роботи.

Мінімальні вимоги до сервера для ефективної роботи системи включають наступне:

- Операційна система: Операційна система класу сервера, така як Windows Server, Linux (наприклад, Ubuntu Server, CentOS) або FreeBSD.
- Процесор: Мінімум чотирьохядерний процесор з тактовою частотою не менше 2,0 ГГц.
- Оперативна пам'ять (RAM): Мінімум 8 ГБ оперативної пам'яті.
- Сховище даних: Мінімум 500 ГБ вільного простору на жорсткому диску або SSD для зберігання додаткової інформації та даних користувачів.
- Мережеві можливості: Підтримка Ethernet або Wi-Fi для з'єднання з локальною мережею та Інтернетом.

- **Безперебійне живлення:** Забезпечення стабільного електропостачання для сервера з можливістю автоматичного резервного живлення.
- **Безпека даних:** Використання відповідних заходів для захисту даних, включаючи резервне копіювання, захист від несанкціонованого доступу та інші заходи безпеки.
- **Браузер:** Встановлення останньої версії браузера для віддаленого керування сервером та доступу до адміністративного інтерфейсу.
- **Інше обладнання:** Надійний серверний корпус, система охолодження, мережеві комутатори та інше необхідне обладнання для забезпечення надійності та безпеки сервера.

Схема мережі передачі даних

Схема мережі передачі даних для системи передбачає окремий сервер, який розташований у захищеному і контрольованому приміщенні, відокремленому від робочих станцій. Сервер забезпечує обробку та зберігання даних, керування базою даних, а також виконання основних операцій системи.

Користувачі мають доступ до системи через Інтернет, отримуючи доступ до веб-сайту за допомогою стандартного браузера. За допомогою веб-інтерфейсу користувачі можуть реєструватись у системі, авторизуватись, переглядати інформацію про інструменти, робити запити на користування або повернення інструментів, переглядати повідомлення та інше.

Мережа передачі даних включає у себе інфраструктуру для забезпечення безперебійного з'єднання між сервером та користувачами через Інтернет. Це може включати в себе мережеві пристрої, такі як маршрутизатори, комутатори, мережеві кабелі та бездротові точки доступу, які забезпечують швидку та надійну передачу даних між сервером та робочими станціями.

3.3. Програмне забезпечення

Для розв'язання конкретної задачі комп'ютер має послідовно виконати повністю визначений набір операцій. Ці операції є сукупністю дій, здійснюваних центральним процесором. Потрібні дії та послідовність їх виконання задає програма. Набір програм, що забезпечують можливість

використання комп'ютера для розв'язування різних задач, складають програмне забезпечення комп'ютера[5].

Структура програмного забезпечення нашої інформаційної системи включає кілька ключових компонентів. Перш за все, це базове системне програмне забезпечення, яке дозволяє організувати обробку інформації та забезпечує робоче середовище для прикладних програм. До складу цього компонента входять операційна система Windows 10, а також середовище виконання PHP, яке використовується для веб-сервера. Вибір операційної системи та інструментів розробки базується на їх популярності, надійності та сумісності з іншими компонентами системи.

Для підтримки роботи сервера, на якому функціонує наша інформаційна система обліку будівельних інструментів, я використовую програмне забезпечення MAMP.

MAMP — це безкоштовний веб-сервер для ПК Windows, до складу якого входять дистрибутиви Apache і MySQL. Цей продукт надає розробникам базові інструменти для створення, тестування та керування веб-додатками або веб-сторінками.

Завдяки інтуїтивно зрозумілому інтерфейсу користувачеві легко запускати і зупиняти сервери, змінювати використовувані порти, вибирати версію поширення PHP і кеш, керувати веб-сервером і базами даних MySQL. MAMP — це зручний безкоштовний спосіб тестування веб-сторінок і веб-додатків локально, без фактичного доступу до Інтернету. Якщо ж вам необхідний більш просунутий функціонал, можете купити версію PRO[6].

Отже MAMP поєднує в собі такі компоненти, як веб-сервер Apache, базу даних MySQL та інтерпретатор мови програмування. Цей набір програм забезпечує стабільну роботу сервера та дозволяє швидко запустити та налаштувати середовище для розробки веб-додатків. Зручно використовувати це програмне забезпечення для розробки проекту, адже воно забезпечує доступ до всіх необхідних можливостей.

Завдяки MAMP можемо легко і ефективно розробляти, тестувати та вдосконалювати нашу інформаційну систему, переконуючись в її

працездатності та надійності перед релізом виробу в продукцію. Таке програмне забезпечення спрощує робочий процес, забезпечуючи зручне та надійне середовище для розробки.

В процесі розробки інформаційної системи обліку будівельних інструментів використовувалось інтегроване середовище розробки (IDE) Visual Studio Code.

Visual Studio Code — це легкий, але потужний редактор вихідного коду, який працює на робочому столі та доступний для Windows, macOS і Linux. Він поставляється з вбудованою підтримкою JavaScript, TypeScript і Node.js і має багату екосистему розширень для інших мов і середовищ виконання (таких як C++, C#, Java, Python, PHP, Go, .NET)[7].

Однією з основних переваг Visual Studio Code є його легкість використання та налаштування. Він має інтуїтивний і зручний інтерфейс, який дозволяє швидко зорієнтуватися в коді та легко виконувати різноманітні завдання.

3.4. Результати реалізації інформаційної системи

Результатом виконаної роботи є спроектована та розроблена інформаційна система обліку будівельних інструментів, яка була розроблена з метою автоматизації процесів обліку, управління та моніторингу інструментів, що використовуються на будівельних майданчиках. Вона забезпечує ефективний контроль над інструментами, полегшує обробку запитів на інструменти та забезпечує оперативне сповіщення користувачів про важливі події.

Реалізовані функціональні можливості

Сторінки зі списком інструментів, користувачів, об'єктів та повідомлень:

- **Список інструментів:** Користувачі можуть переглядати всі доступні інструменти, їх статус та місцезнаходження. Інтерфейс дозволяє легко знаходити потрібний інструмент завдяки функції пошуку та фільтрації за різними параметрами.

- **Список користувачів:** Адміністратори можуть переглядати та керувати обліковими записами всіх користувачів системи. Цей функціонал включає можливість додавання, редагування та видалення користувачів.

- **Список об'єктів:** Всі будівельні об'єкти, на яких використовуються інструменти, зібрані у спеціальному списку. Кожен об'єкт має детальний опис, що дозволяє швидко визначити, де і які інструменти використовуються.

- **Список повідомлень:** Система відображає всі повідомлення, що стосуються різних подій, пов'язаних з інструментами та користувачами. Це дозволяє користувачам бути в курсі всіх важливих змін та подій.

Відправлення та обробка запитів на інструменти:

- Користувачі можуть створювати запити на видачу або повернення інструментів. Ці запити автоматично надсилаються відповідальним особам для подальшої обробки.

- Система обробляє запити в режимі реального часу, сповіщаючи користувачів про статус їхніх запитів. Це забезпечує швидке та ефективне управління інструментами.

Додавання коментарів:

- Користувачі можуть залишати коментарі до запитів та інструментів. Це дозволяє фіксувати важливі деталі та зауваження, що стосуються використання інструментів.

- Коментарі зберігаються в системі і доступні для перегляду всіма відповідальними особами, що сприяє кращій комунікації та взаєморозумінню між користувачами.

Сповіщення користувачів:

- Система автоматично надсилає повідомлення користувачам у разі важливих подій. Наприклад, коли користувач створює запит на інструмент, відповідальний адміністратор отримує повідомлення про новий запит.

- Користувачі також отримують сповіщення про нові коментарі до їхніх запитів або про зміну статусу інструментів, що вони використовують. Це забезпечує постійний зв'язок між користувачами та адміністраторами системи.

Результати тестування

Хоча наша система не була впроваджена на реальних підприємствах, ми здійснили всебічне тестування в умовах, наближених до реальних. Для тестування ми використовували контрольний набір даних, що включав інформацію про інструменти, користувачів та об'єкти. Результати тестування показали, що система успішно виконує всі заявлені функції та забезпечує високу ефективність управління інструментами.

В процесі тестування система виявила себе як надійний інструмент для автоматизації процесів обліку інструментів. Зручність інтерфейсу, швидкість обробки запитів та своєчасність сповіщень є плюсами цієї системи. Всі функціональні можливості системи працювали стабільно, без значних помилок або збоїв.

Потенційна практична цінність та економічна ефективність

Результати тестування свідчать про значний потенціал системи для реального використання на будівельних майданчиках. Впровадження нашої інформаційної системи обліку будівельних інструментів дозволить суттєво знизити час на обробку запитів та підвищити ефективність управління інструментами. Це, в свою чергу, може призвести до зниження витрат на обслуговування інструментів та підвищення загальної продуктивності працівників.

Система забезпечує високий рівень автоматизації, що дозволяє зменшити кількість рутинних завдань для співробітників, звільняючи їх час для виконання більш важливих робіт. Крім того, своєчасне сповіщення про події мінімізує ризик втрати інструментів або їх неправильного використання.

Рекомендації щодо подальших досліджень та удосконалень

На основі проведеного аналізу ми рекомендуємо продовжити дослідження в напрямку інтеграції системи з іншими інформаційними системами підприємства, що дозволить створити єдину інформаційну платформу для управління всіма ресурсами компанії. Також варто розглянути можливість впровадження мобільних додатків для ще більшої зручності

користувачів, що дозволить забезпечити доступ до системи з будь-якого місця та в будь-який час.

Подальші дослідження можуть також включати розробку нових функцій, таких як автоматичне замовлення нових інструментів у разі їх зношення або втрати, інтеграцію з системами планування проектів та аналізу продуктивності працівників. Це дозволить зробити систему ще більш корисною та багатофункціональною.

Таким чином, наша інформаційна система обліку будівельних інструментів має значний потенціал для реального використання на підприємствах. Вона забезпечує ефективне управління інструментами, полегшує процеси обліку та моніторингу, підвищує продуктивність працівників та знижує витрати на обслуговування інструментів.

Якщо користувач ще не має облікового запису він має скористатись сторінкою реєстрації, яка показана на рисунку 3.9.

Рисунок 3.9. Сторінка реєстрації

Зі сторінки реєстрації користувач може перейти на сторінку входу в існуючий обліковий запис.

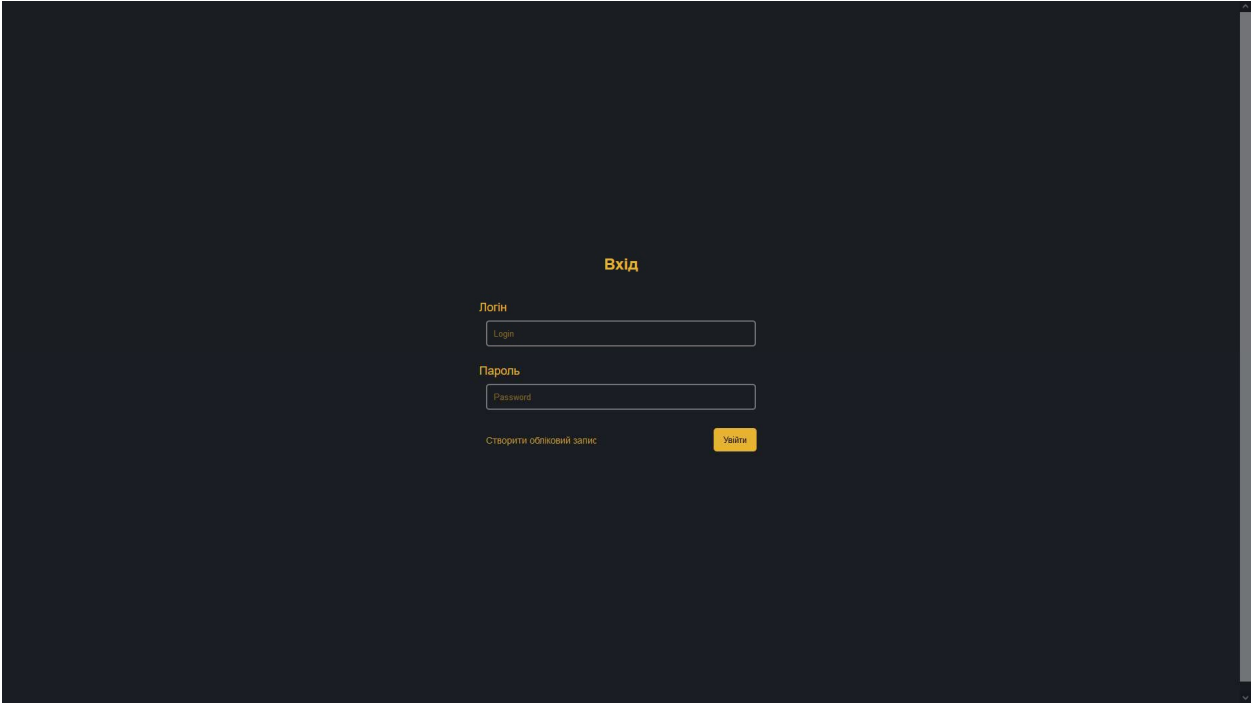


Рисунок 3.10. Сторінка входу

На цьому рисунку показано сторінку зі списком всіх доступних інструментів. Кожен інструмент має детальний опис, статус та місцезнаходження. Користувачі можуть використовувати пошук та фільтрацію для швидкого знаходження необхідного інструменту.

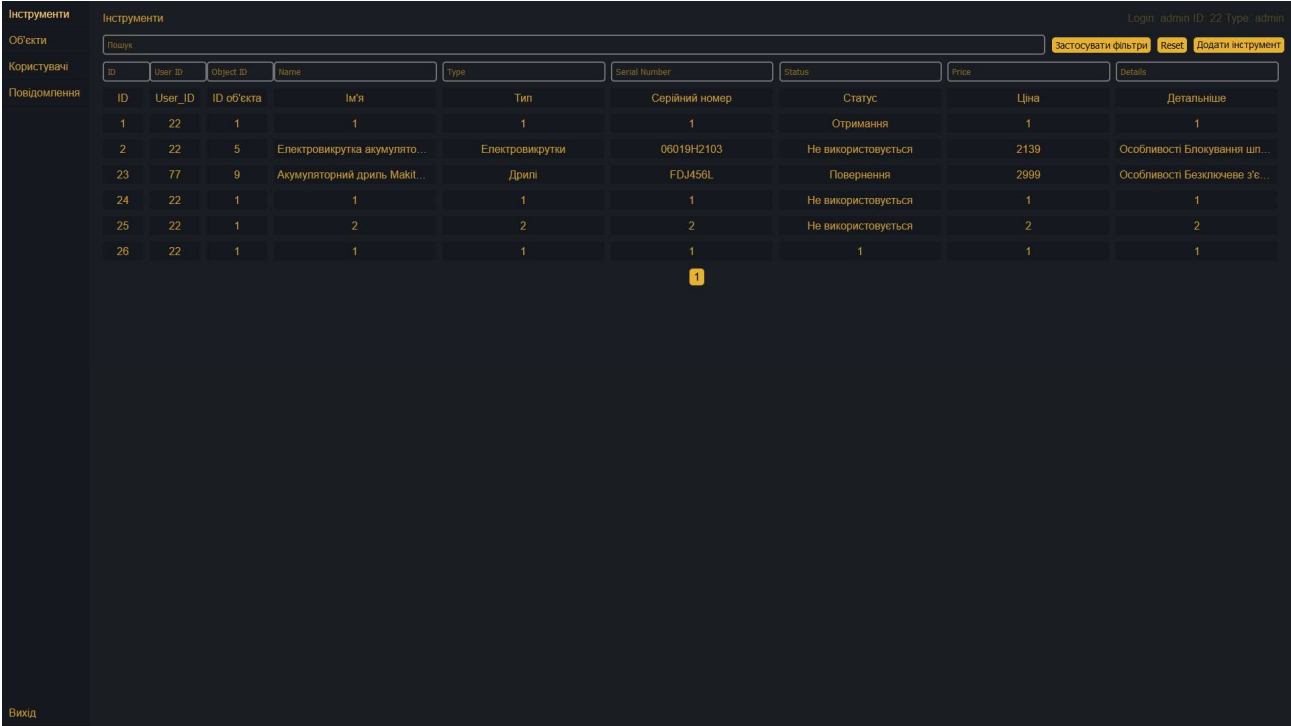


Рисунок 3.11. Сторінка “Інструменти”

Сторінка детальної інформації про інструмент є ключовою складовою інформаційної системи обліку будівельних інструментів. Ця сторінка надає

користувачам повний доступ до всіх важливих даних про конкретний інструмент, забезпечуючи всебічне розуміння його поточного стану, історії використання та доступності.

На цій сторінці користувачі можуть знайти детальний опис інструмента, включаючи його технічні характеристики, статус, місцезнаходження та поточного користувача. Це дозволяє швидко оцінити стан інструмента та визначити його доступність для подальшого використання.

Крім базової інформації, сторінка також дозволяє користувачам додавати та читати коментарі. Коментарі можуть містити важливі зауваження та відгуки щодо використання інструмента, його технічного стану чи будь-які інші корисні деталі. Це сприяє кращій комунікації між користувачами та адміністраторами, а також допомагає зберігати всю важливу інформацію в одному місці.

Історія користування інструментом є ще однією важливою функцією цієї сторінки. Вона дозволяє переглядати всі попередні запити на інструмент, включаючи дати видачі та повернення, імена користувачів. Це забезпечує повний контроль над життєвим циклом інструмента.

Сторінка також надає можливість відправляти та обробляти запити на користування інструментом. Користувачі можуть створювати нові запити. Адміністратори, в свою чергу, можуть обробляти ці запити, підтверджуючи або відхиляючи їх, а також вносити відповідні зміни до статусу інструмента. Це забезпечує ефективне управління доступністю інструментів та мінімізує час на обробку запитів.

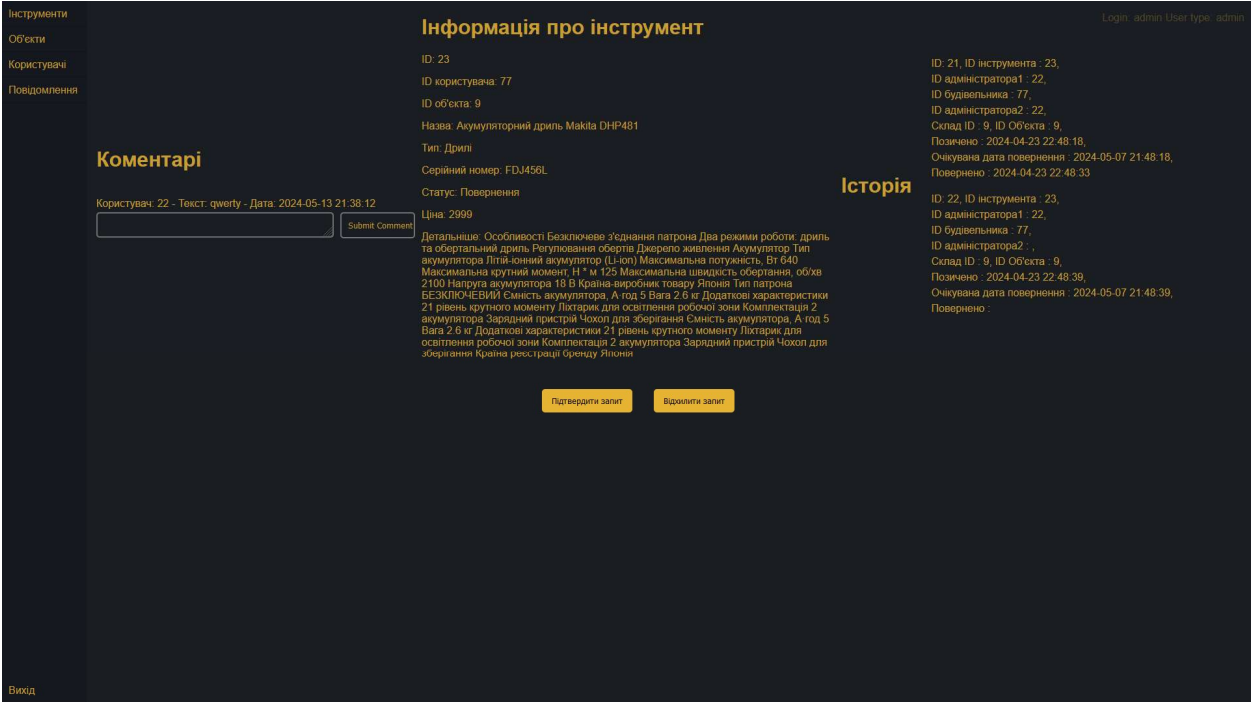


Рисунок 3.12. Сторінка “Інформація про інструмент”

На сторінці зі списком користувачів представлений перелік усіх користувачів системи разом з їхніми основними контактними даними та ролями у системі.

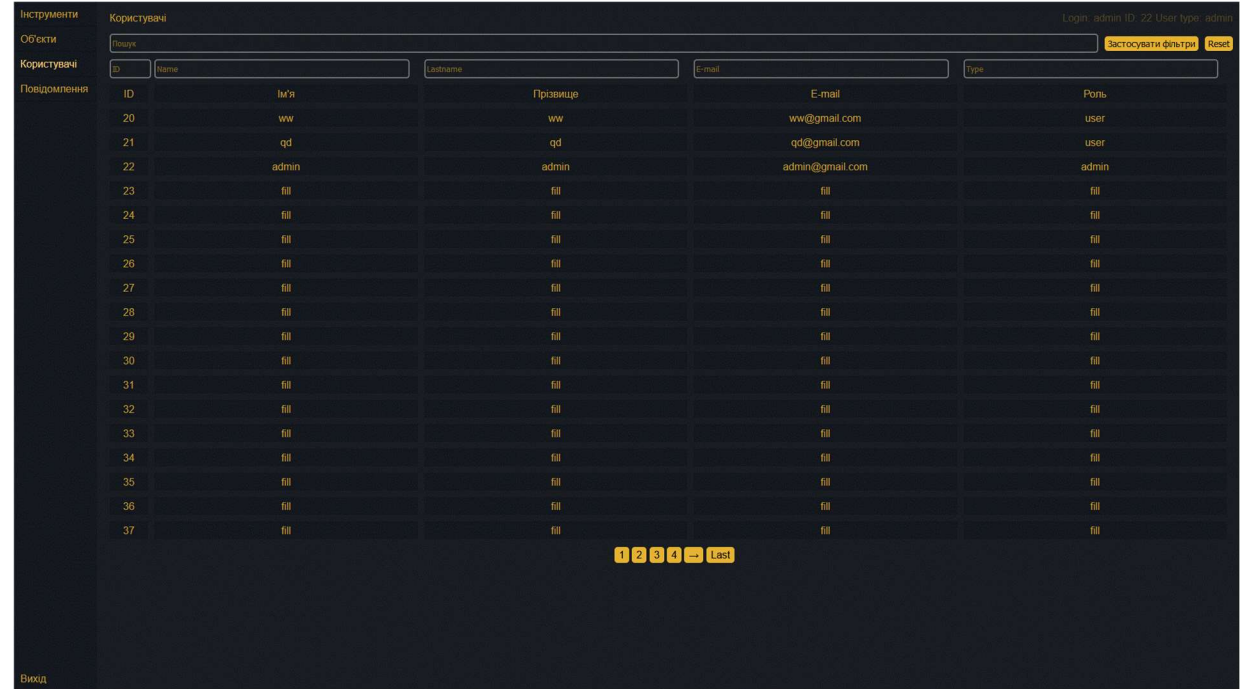


Рисунок 3.13. Сторінка “Користувачі”

Цей рисунок відображає сторінку зі списком всіх повідомлень, які були надіслані користувачу. Повідомлення включають інформацію про нові запити, коментарі та інші важливі події.

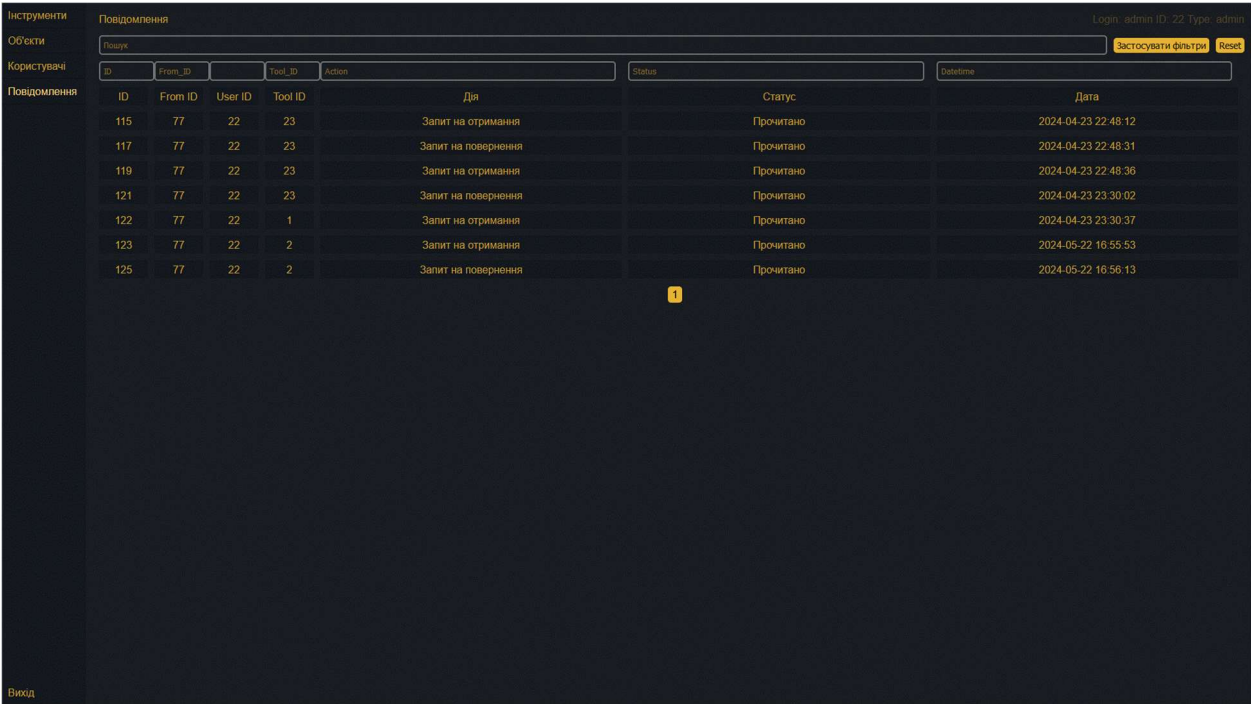


Рисунок 3.14. Сторінка “Повідомлення”

На цьому рисунку показана сторінка, де зібрані всі будівельні об’єкти, пов’язані з системою. Є детальна інформація про об’єкти та посилання на місцезнаходження кожного з них.

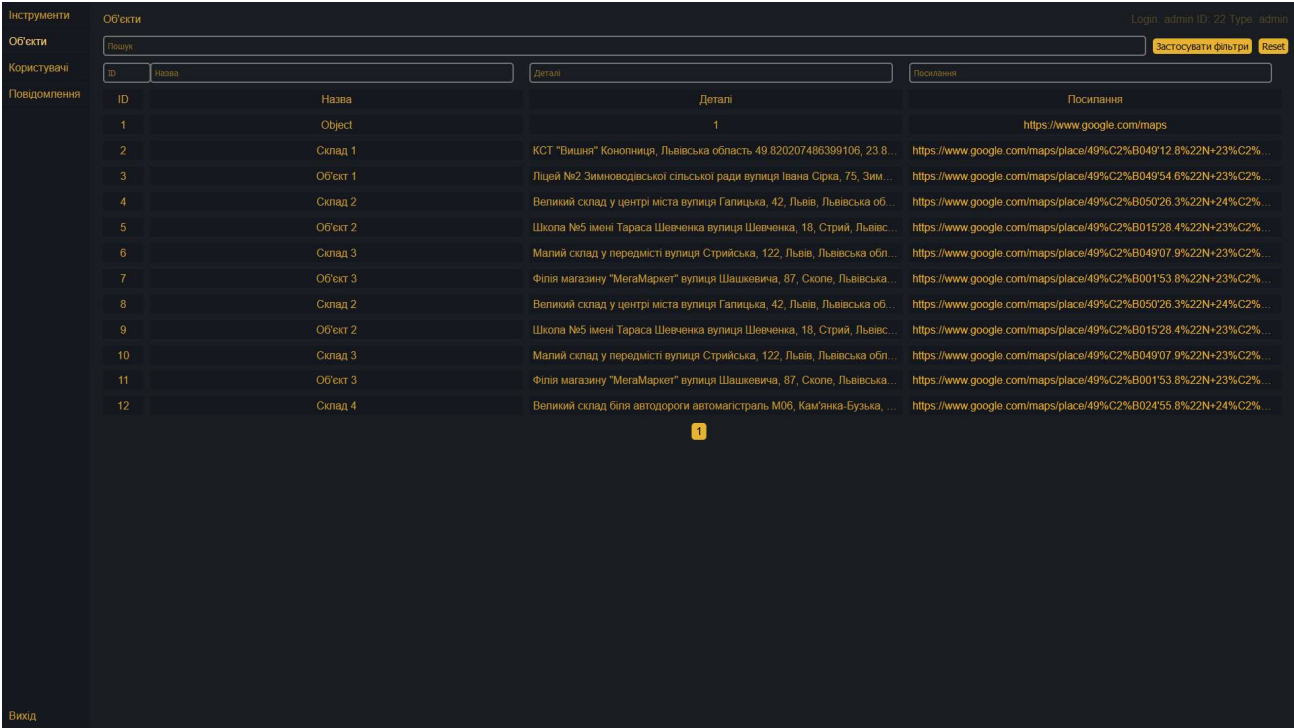


Рисунок 3.15. Сторінка “Об’єкти”

Сторінка додавання інструментів призначена виключно для працівників складу, забезпечуючи їм можливість вводити нові інструменти в систему.

Доступ до цієї сторінки обмежено для забезпечення контролю за введенням даних і підтриманням актуальності інформації в системі.

На цій сторінці працівник складу може ввести всі необхідні дані про новий інструмент. Це включає назву інструмента, його технічні характеристики, серійний номер та іншу інформацію. Детальний опис інструмента допомагає іншим користувачам швидко знайти потрібний інструмент та зрозуміти його основні властивості.

The screenshot shows a web application interface with a dark theme. On the left is a vertical sidebar with navigation links: 'Інструменти' (Instruments), 'Об'єкти' (Objects), 'Користувачі' (Users), 'Повідомлення' (Notifications), and 'Вихід' (Logout). The main content area is titled 'Додати інструмент' (Add Instrument). It contains several input fields with placeholder text: 'Назва' (Name), 'Тип' (Type), 'Серійний номер' (Serial number), 'Статус' (Status), 'Ціна' (Price), 'ID об'єкта' (Object ID), and 'Детальніше' (More details). A yellow button labeled 'Додати інструмент' (Add Instrument) is at the bottom of the form. In the top right corner, there is a login status indicator: 'Login: admin User type: admin'.

Рисунок 3.16. Сторінка “Додати інструмент”

Фрагменти коду інформаційної системи наведено у додатках 3-7.

ВИСНОВКИ

У процесі виконання кваліфікаційного бакалаврського проекту на тему "Проектування інформаційної системи обліку будівельних інструментів" була досягнута головна мета – спроектовано та розроблено ефективну систему для автоматизації обліку, моніторингу та управління будівельними інструментами. Усі поставлені завдання були успішно виконані, а результати роботи підтверджують доцільність використання розробленої системи на практиці.

Створено інформаційну систему обліку будівельних інструментів, яка дозволяє ефективно управляти інструментами на складі, відстежувати їх використання, а також полегшує взаємодію між працівниками через систему повідомлень і запитів. Це значно покращує процес управління ресурсами та дозволяє уникнути багатьох проблем, пов'язаних з ручним веденням обліку, таких як втрата інформації, помилки в записах та затримки в обробці даних.

Реалізовано ключові функціональні компоненти системи: сторінки зі списком інструментів, користувачів, об'єктів, повідомлень, відправлення та обробка запитів на інструменти, додавання коментарів, а також сповіщення користувачів про важливі події. Завдяки цьому забезпечується високий рівень прозорості та контрольованості процесів, що відбуваються в системі. Користувачі можуть швидко і легко знаходити необхідну інформацію, що значно підвищує продуктивність праці.

Забезпечено автоматичне сповіщення користувачів про події, що їх стосуються, завдяки чому покращено оперативність і точність передачі інформації. Це дозволяє зменшити кількість помилок і покращити продуктивність праці. Користувачі оперативно отримують повідомлення про нові запити, коментарі або зміни в статусі інструментів, що дозволяє їм швидко реагувати на ситуації та приймати обґрунтовані рішення.

Використано сучасні технології для розробки системи: середовище розробки Visual Studio Code для написання коду, програма MAMP для запуску сервера та бази даних MySQL, а також мови програмування PHP, HTML та CSS для створення інтерфейсу користувача. Завдяки цьому система вийшла

стабільною, надійною і зручною у використанні. Вибір цих технологій обумовлений їх високою продуктивністю, гнучкістю і широкими можливостями для розширення функціоналу в майбутньому.

Система забезпечує можливість додавання та читання коментарів, перегляду історії користування інструментами та обробки запитів на їх використання. Це сприяє більшій прозорості і відстежуваності операцій з інструментами, що підвищує контроль і відповідальність працівників. Кожен інструмент має свою історію, що дозволяє легко відстежувати його використання та вчасно проводити необхідне обслуговування або ремонт.

Реалізовано зручний інтерфейс для додавання нових інструментів в систему, який доступний лише працівникам складу, що забезпечує безпеку даних та захищеність від несанкціонованих дій. Ця функція дозволяє швидко і легко додавати нові інструменти в систему, вказуючи їх основні характеристики, що значно спрощує процес обліку і дозволяє уникнути помилок, пов'язаних з ручним введенням даних.

Сторінка зі списком користувачів надає інформацію про електронну пошту та роль кожного користувача в системі, що дозволяє дізнатись інформацію про працівників складу та будівельників.

Сторінка реєстрації нових користувачів дозволяє легко і швидко створити обліковий запис, вводячи необхідні дані, забезпечуючи захищений вхід до системи. Це спрощує процес реєстрації нових користувачів і забезпечує високий рівень безпеки системи.

Сторінка детальної інформації про інструмент надає користувачам можливість отримати повну інформацію про конкретний інструмент, додавати та читати коментарі, переглядати історію використання інструменту, а також відправляти та обробляти запити на його використання. Це забезпечує повну прозорість і контроль за використанням кожного інструменту, що дозволяє підвищити ефективність управління ресурсами.

Розроблена система продемонструвала високу ефективність у тестовому середовищі, показавши значне скорочення часу на обробку запитів і підвищення точності обліку інструментів. Аналіз результатів показав, що

система має високий потенціал для практичного застосування на будівельних об'єктах і складах.

Проведено повний цикл розробки системи: від аналізу вимог і проектування до реалізації та тестування. У процесі роботи були використані сучасні методи програмування та інструменти розробки, що дозволило створити надійний і функціональний програмний продукт. Застосовані рішення забезпечують високу продуктивність системи та її здатність до масштабування в майбутньому.

Подальші дослідження можуть бути спрямовані на розширення функціоналу системи, інтеграцію з іншими інформаційними системами підприємства та оптимізацію продуктивності для роботи з великими обсягами даних. Рекомендується також продовжувати вдосконалювати систему безпеки та захисту даних, щоб забезпечити високий рівень захищеності інформації. Крім того, варто розглянути можливість впровадження мобільного додатку, що дозволить користувачам ще зручніше взаємодіяти із системою і оперативно отримувати доступ до необхідної інформації.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. СКБД MySQL і доступ до БД в PHP [Електронний ресурс]. – Режим доступу: <http://www.znannya.org/?view=mysql-intro>
2. Інфологічна модель даних “Сутність-зв’язок” [Електронний ресурс]. – Режим доступу: <https://studfile.net/preview/7363846/>
3. Проектування фізичної моделі ІС [Електронний ресурс]. – Режим доступу:
https://elearning.sumdu.edu.ua/free_content/lectured:de1c9452f2a161439391120ee-f364dd8ce4d8e5e/20151203140326/204841/index.html
4. Діаграма розгортання: Підручник з UML [Електронний ресурс]. – Режим доступу: <https://www.guru99.com/uk/deployment-diagram-uml-example.html>
5. Програмне забезпечення [Електронний ресурс]. – Режим доступу: <https://step.org.ua/konspekt/pz/tema1>
6. Особливості МАР [Електронний ресурс]. – Режим доступу: <https://biblprog.org.ua/ua/mamp/>
7. Visual Studio Code Getting Started [Електронний ресурс]. – Режим доступу: <https://code.visualstudio.com/docs>

ДОДАТКИ

Додаток 1

Нефункціональні вимоги до системи				
req [package] Нефункціональні вимоги до системи [Нефункціональні вимоги до системи]				
Item	Stereotype	Status	Difficulty	Priority
Адаптивність до різних пристроїв Забезпечення адаптивністю до різних пристроїв, включаючи комп'ютери, планшети, мобільні телефони тощо	requirement	Proposed	Medium	Medium
Забезпечення стабільності та надійності середовища Забезпечення стабільності та надійності середовища, у тому числі операційної системи, баз даних, серверного програмного забезпечення та інших компонентів, що використовуються для розгортання системи.	requirement	Proposed	Medium	Medium
Забезпечення умов тестування Система повинна бути піддатна тестуванню в умовах, які максимально відтворюють реальні умови експлуатації, забезпечуючи відтворюваність тестових сценаріїв і надійність результатів тестування.	requirement	Proposed	Medium	Medium
Керування конфігурацією У проєктованій системі потрібно передбачити можливість внесення змін і предствановлені конфігурації. Внесення змін не повинно впливати на процес роботи системи	requirement	Proposed	Medium	Medium
Кросбраузерність Забезпечення коректної роботи сайту на різних браузерах. Забезпечення кросбраузерності	requirement	Proposed	Medium	Medium
Обмежений доступ Доступ до системи є тільки в авторизованих користувачів з необхідними для роботи з системою правами	requirement	Proposed	Medium	Medium
Продуктивність обробки запитів Система повинна забезпечувати можливість обробки не менше 500 запитів на хвилину	requirement	Proposed	Medium	Medium
Розширюваність Система повинна підтримувати збільшення ресурсів (процесора, пам'яті) на існуючих серверах.	requirement	Proposed	Medium	Medium
Спалахозахисність Система не повинна мати великої матеріалів з яскравим миготливим освітленням для безпеки користувачів з епілепсією.	requirement	Proposed	Medium	Medium
Швидке завантаження сторінок Висока швидкість завантаження сторінок та даних (завантаження не довше трьох секунд)	requirement	Proposed	Medium	Medium

Специфікація нефункціональних вимог

БІЗНЕС ВИМОГИ				
req [package] Бізнес вимоги [Бізнес вимоги]				
Item	Stereotype	Status	Difficulty	Priority
Безпека та доступ до даних Система повинна забезпечувати захищений доступ до інформації про будівельні інструменти та обмежувати права доступу до цієї інформації відповідно до ролей та обов'язків користувачів.	requirement	Proposed	Medium	Medium
Відповідальність за інструмент Система повинна забезпечувати облік користувачів відповідальних за інструменти	requirement	Proposed	Medium	Medium
Відстеження руху інструментів Система може фіксувати переміщення інструментів між робочими місцями, складами, проектами ведучи журнал видачі та повернення інструментів.	requirement	Proposed	Medium	Medium
Доступ до актуальної інформації Система повинна забезпечувати швидкий та легкий доступ до актуальної інформації про наявність і стан будівельних інструментів для всіх зацікавлених сторін в організації.	requirement	Proposed	Medium	Medium
Забезпечення безпеки працівників Система повинна допомагати забезпечувати безпеку працівників шляхом відслідковування відповідності будівельних інструментів стандартам безпеки та технічного стану	requirement	Proposed	Medium	Medium
Підтримка мобільності Система повинна бути доступною з різних пристроїв та платформ, щоб користувачі могли отримувати доступ до неї незалежно від місця перебування.	requirement	Proposed	Medium	Medium
Повний облік інвентарю Система повинна забезпечувати повний та точний облік всіх будівельних інструментів, включаючи їхнє розташування, стан, історію використання.	requirement	Proposed	Medium	Medium

Специфікація бізнес вимог

Функціональні вимоги

← req [package] Функціональні вимоги [Функціональні вимоги]

Item	Stereotype	Status	Difficulty	Priority
☒ Авторизація	requirement	Proposed	High	High
☒ Вихід	requirement	Proposed	Low	Medium
☒ Додати коментар	requirement	Proposed	Medium	Medium
☒ Запит на користування інструментом	requirement	Proposed	Medium	High
☒ Запит на повернення інструмента	requirement	Proposed	High	High
☒ Запити	requirement	Proposed	Medium	Medium
☒ Історія	requirement	Proposed	High	Medium
☒ Керування доступом	requirement	Proposed	High	High
☒ Коментарі	requirement	Proposed	Medium	Low
☒ Меню навігації	requirement	Proposed	Low	Medium
☒ Обробка запиту на користування інструментом	requirement	Proposed	High	High
☒ Обробка запиту на повернення інструмента	requirement	Proposed	High	High
☒ Пагінація	requirement	Proposed	Medium	Medium
☒ Перехід на Google maps	requirement	Proposed	Low	Medium
☒ Посилання на сторінку "Детальна інформація про інструмент"	requirement	Proposed	Medium	High
☒ Посилання на сторінку з інструментами користувача	requirement	Proposed	Medium	Medium
☒ Пошук	requirement	Proposed	Medium	Medium
☒ Редагувати коментар	requirement	Proposed	Medium	Medium
☒ Реєстрація	requirement	Proposed	High	High
☒ Сторінка "Детальна інформація про інструмент"	requirement	Proposed	Medium	High
☒ Сторінка "Додати інструмент"	requirement	Proposed	Medium	High
☒ Сторінка "Інструменти"	requirement	Proposed	Medium	High
☒ Сторінка "Користувачі"	requirement	Proposed	Medium	High
☒ Сторінка "Об'єкти"	requirement	Proposed	Medium	High
☒ Сторінка "Повідомлення"	requirement	Proposed	Medium	High
☒ Сторінка "Редагувати дані про інструмент"	requirement	Proposed	Medium	Medium
☒ Фільтри	requirement	Proposed	Medium	Medium

Специфікація функціональних вимог

Додаток 2

Експортотвана БД мовою SQL:

```
-- phpMyAdmin SQL Dump
-- version 5.1.2
-- https://www.phpmyadmin.net/
--
-- Host: localhost:3306
-- Generation Time: May 18, 2024 at 05:09 PM
-- Server version: 5.7.24
-- PHP Version: 7.4.16

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
START TRANSACTION;
SET time_zone = "+00:00";

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8mb4 */;

--
-- Database: `bd1`
--

-----

--
-- Table structure for table `comment`
--

CREATE TABLE `comment` (
  `ID` int(11) NOT NULL,
  `User_ID` int(11) NOT NULL,
  `Tool_ID` int(11) NOT NULL,
  `Text` varchar(255) NOT NULL,
  `Datetime` datetime NOT NULL DEFAULT CURRENT_TIMESTAMP
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

--
-- Dumping data for table `comment`
--

INSERT INTO `comment` (`ID`, `User_ID`, `Tool_ID`, `Text`, `Datetime`) VALUES
(1, 22, 2, " '2024-05-09 17:15:05'",
(2, 22, 23, 'qwerty', '2024-05-13 21:38:12');

-----

--
-- Table structure for table `history`
--

CREATE TABLE `history` (
  `ID` int(11) NOT NULL,
  `Tool_ID` int(11) NOT NULL,
  `Whm1_ID` int(11) NOT NULL,
  `Bldr_ID` int(11) NOT NULL,
  `Whm2_ID` int(11) DEFAULT NULL,
```

```

`Storage_ID` int(11) NOT NULL,
`Object_ID` int(11) DEFAULT NULL,
`Borrow` datetime NOT NULL DEFAULT CURRENT_TIMESTAMP,
`MustReturn` datetime DEFAULT NULL,
`Returned` datetime DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

--
-- Dumping data for table `history`
--

INSERT INTO `history` (`ID`, `Tool_ID`, `Whm1_ID`, `Bldr_ID`, `Whm2_ID`, `Storage_ID`, `Object_ID`,
`Borrow`, `MustReturn`, `Returned`) VALUES
(21, 23, 22, 77, 22, 9, 9, '2024-04-23 22:48:18', '2024-05-07 21:48:18', '2024-04-23 22:48:33'),
(22, 23, 22, 77, NULL, 9, 9, '2024-04-23 22:48:39', '2024-05-07 21:48:39', NULL);

-----

--
-- Table structure for table `notification`
--

CREATE TABLE `notification` (
  `ID` int(11) NOT NULL,
  `From_ID` int(11) NOT NULL,
  `User_ID` int(11) NOT NULL,
  `Tool_ID` int(11) NOT NULL,
  `Action` varchar(32) NOT NULL,
  `Status` varchar(20) NOT NULL DEFAULT 'Не прочитано',
  `Datetime` datetime NOT NULL DEFAULT CURRENT_TIMESTAMP
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

--
-- Dumping data for table `notification`
--

INSERT INTO `notification` (`ID`, `From_ID`, `User_ID`, `Tool_ID`, `Action`, `Status`, `Datetime`)
VALUES
(115, 77, 22, 23, 'Запит на отримання', 'Прочитано', '2024-04-23 22:48:12'),
(116, 22, 77, 23, 'Підтверджено отримання', 'Прочитано', '2024-04-23 22:48:18'),
(117, 77, 22, 23, 'Запит на повернення', 'Прочитано', '2024-04-23 22:48:31'),
(118, 22, 77, 23, 'Підтверджено повернення', 'Прочитано', '2024-04-23 22:48:33'),
(119, 77, 22, 23, 'Запит на отримання', 'Прочитано', '2024-04-23 22:48:36'),
(120, 22, 77, 23, 'Підтверджено отримання', 'Прочитано', '2024-04-23 22:48:39'),
(121, 77, 22, 23, 'Запит на повернення', 'Прочитано', '2024-04-23 23:30:02'),
(122, 77, 22, 1, 'Запит на отримання', 'Прочитано', '2024-04-23 23:30:37');

-----

--
-- Table structure for table `object`
--

CREATE TABLE `object` (
  `ID` int(11) NOT NULL,
  `Name` varchar(32) NOT NULL DEFAULT 'Object',
  `Details` varchar(255) DEFAULT NULL,
  `Link` varchar(255) NOT NULL DEFAULT 'https://www.google.com/maps'
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

--

```

-- Dumping data for table `object`

--

```
INSERT INTO `object` (`ID`, `Name`, `Details`, `Link`) VALUES
(1, 'Object', '1', 'https://www.google.com/maps'),
(2, 'Склад 1', 'КСТ  \\'Вишня\\'\\r\\nКонопниця, Львівська область\\r\\n49.820207486399106,
23.86135712972616',
'https://www.google.com/maps/place/49%C2%B049\\'12.8%22N+23%C2%B051\\'40.9%22E/@49.8202109,23.8587
822,17z/data=!3m1!4b1!4m4!3m3!8m2!3d49.8202075!4d23.8613571?hl=uk-UK&entry=ttu'),
(3, 'Об'єкт 1', 'Ліцей №2 Зимноводівської сільської ради\\r\\nвулиця Івана Сірка, 75, Зимна Вода,
Львівська область, 81110\\r\\n49.831857292794375, 23.870375468180896',
'https://www.google.com/maps/place/49%C2%B049\\'54.6%22N+23%C2%B052\\'13.4%22E/@49.8318422,23.8693
706,18z/data=!3m1!4b1!4m4!3m3!8m2!3d49.8318408!4d23.8703925?hl=uk-UK&entry=ttu'),
(4, 'Склад 2', 'Великий склад у центрі міста\\r\\nвулиця Галицька, 42, Львів, Львівська область,
79000\\r\\n49.8406255230776, 24.029892126662073',
'https://www.google.com/maps/place/49%C2%B050\\'26.3%22N+24%C2%B001\\'47.6%22E/@49.8406294,24.0294
105,17z/data=!3m1!4b1!4m4!3m3!8m2!3d49.8406255!4d24.0313513?hl=uk-UK&entry=ttu'),
(5, 'Об'єкт 2', 'Школа №5 імені Тараса Шевченка\\r\\nвулиця Шевченка, 18, Стрий, Львівська область,
82400\\r\\n49.25788215058098, 23.846052772666943',
'https://www.google.com/maps/place/49%C2%B015\\'28.4%22N+23%C2%B050\\'45.8%22E/@49.2578793,23.8460
514,17z/data=!3m1!4b1!4m4!3m3!8m2!3d49.2578822!4d23.8472401?hl=uk-UK&entry=ttu'),
(6, 'Склад 3', 'Малий склад у передмісті\\r\\nвулиця Стрийська, 122, Львів, Львівська область,
79040\\r\\n49.81887548471962, 23.97301008956813',
'https://www.google.com/maps/place/49%C2%B049\\'07.9%22N+23%C2%B058\\'22.8%22E/@49.8188794,23.9735
792,17z/data=!3m1!4b1!4m4!3m3!8m2!3d49.8188755!4d23.9757683?hl=uk-UK&entry=ttu'),
(7, 'Об'єкт 3', 'Філія магазину \\'МегаМаркет\\'\\r\\nвулиця Шашкевича, 87, Сколе, Львівська область,
82600\\r\\n49.031620963783956, 23.511738334269603',
'https://www.google.com/maps/place/49%C2%B001\\'53.8%22N+23%C2%B030\\'42.3%22E/@49.0316203,23.5095
492,17z/data=!3m1!4b1!4m4!3m3!8m2!3d49.031621!4d23.511738?hl=uk-UK&entry=ttu'),
(8, 'Склад 2', 'Великий склад у центрі міста\\r\\nвулиця Галицька, 42, Львів, Львівська область,
79000\\r\\n49.8406255230776, 24.029892126662073',
'https://www.google.com/maps/place/49%C2%B050\\'26.3%22N+24%C2%B001\\'47.6%22E/@49.8406294,24.0294
105,17z/data=!3m1!4b1!4m4!3m3!8m2!3d49.8406255!4d24.0313513?hl=uk-UK&entry=ttu'),
(9, 'Об'єкт 2', 'Школа №5 імені Тараса Шевченка\\r\\nвулиця Шевченка, 18, Стрий, Львівська область,
82400\\r\\n49.25788215058098, 23.846052772666943',
'https://www.google.com/maps/place/49%C2%B015\\'28.4%22N+23%C2%B050\\'45.8%22E/@49.2578793,23.8460
514,17z/data=!3m1!4b1!4m4!3m3!8m2!3d49.2578822!4d23.8472401?hl=uk-UK&entry=ttu'),
(10, 'Склад 3', 'Малий склад у передмісті\\r\\nвулиця Стрийська, 122, Львів, Львівська область,
79040\\r\\n49.81887548471962, 23.97301008956813',
'https://www.google.com/maps/place/49%C2%B049\\'07.9%22N+23%C2%B058\\'22.8%22E/@49.8188794,23.9735
792,17z/data=!3m1!4b1!4m4!3m3!8m2!3d49.8188755!4d23.9757683?hl=uk-UK&entry=ttu'),
(11, 'Об'єкт 3', 'Філія магазину \\'МегаМаркет\\'\\r\\nвулиця Шашкевича, 87, Сколе, Львівська область,
82600\\r\\n49.031620963783956, 23.511738334269603',
'https://www.google.com/maps/place/49%C2%B001\\'53.8%22N+23%C2%B030\\'42.3%22E/@49.0316203,23.5095
492,17z/data=!3m1!4b1!4m4!3m3!8m2!3d49.031621!4d23.511738?hl=uk-UK&entry=ttu'),
(12, 'Склад 4', 'Великий склад біля автодороги\\r\\nавтомагістраль М06, Кам'янка-Бузька, Львівська
область, 80302\\r\\n49.41551280190733, 24.37092668979777',
'https://www.google.com/maps/place/49%C2%B024\\'55.8%22N+24%C2%B022\\'15.3%22E/@49.4155122,24.3709
234,17z/data=!3m1!4b1!4m4!3m3!8m2!3d49.415513!4d24.372751?hl=uk-UK&entry=ttu');
```

-- -----

--

-- Table structure for table `tool`

--

```
CREATE TABLE `tool` (
  `ID` int(11) NOT NULL,
  `User_ID` int(11) NOT NULL,
  `Object_ID` int(11) NOT NULL,
  `Name` varchar(64) NOT NULL,
```

```

`Type` varchar(32) DEFAULT NULL,
`SerialNumber` varchar(64) DEFAULT NULL,
`Status` varchar(20) NOT NULL,
`Price` float DEFAULT NULL,
`Image` varchar(255) DEFAULT NULL,
`Details` text
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

--
-- Dumping data for table `tool`
--

INSERT INTO `tool` (`ID`, `User_ID`, `Object_ID`, `Name`, `Type`, `SerialNumber`, `Status`, `Price`,
`Image`, `Details`) VALUES
(1, 22, 1, 'I', 'I', 'Отримання', 1, 'I', 'I'),
(2, 22, 5, 'Електровикрутка акумуляторна Bosch Professional GO 2', 'Електровикрутки', '06019H2103',
'Не використовується', 2139, NULL, 'Особливості\r\n Блокування шпинделя\r\n Реверс\r\n
Регулювання крутного моменту\r\nДжерело живлення\r\n Акумулятор\r\nТип акумулятора\r\n Літій-
іонний акумулятор (Li-ion)\r\nМаксимальний крутний момент, Н * м\r\n 5\r\nЧас заряду акумулятора,
год\r\n 1.5\r\nМаксимальна швидкість обертання, об/хв\r\n 360\r\nРежимів крутного моменту\r\n
5\r\nРежим роботи\r\n 3 реверсом\r\nНапруга акумулятора\r\n 3.6 В\r\nКраїна-виробник товару\r\n
Малайзія\r\nТип патрона\r\n Шестигранний\r\nЄмність акумулятора, А*год\r\n 1.5\r\nВага\r\n 0.3
кг\r\nДодаткові характеристики\r\n Реверс\r\n Блокування шпинделя\r\nКомплектація\r\n USB
кабель\r\n Інструкція з експлуатації\r\n Пластиковий кейс\r\nКраїна реєстрації бренду\r\n
Німеччина'),
(23, 77, 9, 'Акумуляторний дріль Makita DHP481', 'Дрилі', 'FDJ456L', 'Повернення', 2999, NULL,
'Особливості\r\n Безключеве з'єднання патрона\r\n Два режими роботи: дріль та обертальний дріль\r\n
Регулювання обертів\r\nДжерело живлення\r\n Акумулятор\r\nТип акумулятора\r\n Літій-іонний акумулятор
(Li-ion)\r\nМаксимальна потужність, Вт\r\n 640\r\nМаксимальна крутний момент, Н * м\r\n
125\r\nМаксимальна швидкість обертання, об/хв\r\n 2100\r\nНапруга акумулятора\r\n 18 В\r\nКраїна-виробник
товару\r\n Японія\r\nТип патрона\r\n БЕЗКЛЮЧЕВИЙ\r\nЄмність акумулятора, А*год\r\n 5\r\nВага\r\n 2.6
кг\r\nДодаткові характеристики\r\n 21 рівень крутного моменту\r\n Ліхтарик для освітлення робочої
зони\r\nКомплектація\r\n 2 акумулятора\r\n Зарядний пристрій\r\n Чохол для зберігання\r\nЄмність
акумулятора, А*год\r\n 5\r\nВага\r\n 2.6 кг\r\nДодаткові характеристики\r\n 21 рівень крутного моменту\r\n
Ліхтарик для освітлення робочої зони\r\nКомплектація\r\n 2 акумулятора\r\n Зарядний пристрій\r\n Чохол для
зберігання\r\nКраїна реєстрації бренду\r\n Японія'),
(24, 22, 1, 'I', 'I', 'I', 'Не використовується', 1, 'I', 'I'),
(25, 22, 1, '2', '2', '2', 'Не використовується', 2, '2', '2');

-----

--
-- Table structure for table `user`
--

CREATE TABLE `user` (
`ID` int(11) NOT NULL,
`Name` varchar(20) NOT NULL,
`Lastname` varchar(20) NOT NULL,
`E_mail` varchar(50) NOT NULL,
`Login` varchar(50) NOT NULL,
`Password` varchar(255) NOT NULL,
`Type` varchar(32) NOT NULL DEFAULT 'user'
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

--
-- Dumping data for table `user`
--

INSERT INTO `user` (`ID`, `Name`, `Lastname`, `E_mail`, `Login`, `Password`, `Type`) VALUES

```

```

(20, 'ww', 'ww', 'ww@gmail.com', 'ww',
'$2y$10$wU.7RVU7ItpQZWnC6CqBeOX.T/fJEsijiTWxoIMbUWj0rIZ6R55s6', 'user'),
(21, 'qd', 'qd', 'qd@gmail.com', 'qd',
'$2y$10$a/6T9jI9x0ZoKz3j0.u0UuC7TZ1sAwAbUBDKneOdqL/vN78p62cQC', 'user'),
(22, 'admin', 'admin', 'admin@gmail.com', 'admin',
'$2y$10$12m9ID2cEUV0.VPJOWv6UO/mak6LxJ.4FvscD.s0BjpDCqoKip5SW', 'admin'),
(23, 'fill', 'fill', 'fill', 'fill', 'fill'),
(76, 'fill', 'fill', 'fill', 'fill', 'fill'),
(77, 'user', 'user', 'user@gmail.com', 'user',
'$2y$10$ScWn7wr4d3uZjg7NRkTpM0ea5qkneZIElqw8TtbBxWeN/2pimy.QVC', 'user');

```

--

-- Indexes for dumped tables

--

--

-- Indexes for table `comment`

--

```

ALTER TABLE `comment`
  ADD PRIMARY KEY (`ID`),
  ADD KEY `User_ID` (`User_ID`),
  ADD KEY `Tool_ID` (`Tool_ID`);

```

--

-- Indexes for table `history`

--

```

ALTER TABLE `history`
  ADD PRIMARY KEY (`ID`),
  ADD KEY `Tool_ID` (`Tool_ID`),
  ADD KEY `Whm1_ID` (`Whm1_ID`),
  ADD KEY `Bldr_ID` (`Bldr_ID`),
  ADD KEY `Whm2_ID` (`Whm2_ID`),
  ADD KEY `Storage_ID` (`Storage_ID`),
  ADD KEY `Object_ID` (`Object_ID`);

```

--

-- Indexes for table `notification`

--

```

ALTER TABLE `notification`
  ADD PRIMARY KEY (`ID`),
  ADD KEY `User_ID` (`User_ID`),
  ADD KEY `Tool_ID` (`Tool_ID`),
  ADD KEY `From_ID` (`From_ID`);

```

--

-- Indexes for table `object`

--

```

ALTER TABLE `object`
  ADD PRIMARY KEY (`ID`);

```

--

-- Indexes for table `tool`

--

```

ALTER TABLE `tool`
  ADD PRIMARY KEY (`ID`),
  ADD KEY `User_ID` (`User_ID`) USING BTREE,
  ADD KEY `Object_ID` (`Object_ID`);

```

--

-- Indexes for table `user`

--

```

ALTER TABLE `user`
  ADD PRIMARY KEY (`ID`);

--
-- AUTO_INCREMENT for dumped tables
--

--
-- AUTO_INCREMENT for table `comment`
--
ALTER TABLE `comment`
  MODIFY `ID` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=3;

--
-- AUTO_INCREMENT for table `history`
--
ALTER TABLE `history`
  MODIFY `ID` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=23;

--
-- AUTO_INCREMENT for table `notification`
--
ALTER TABLE `notification`
  MODIFY `ID` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=123;

--
-- AUTO_INCREMENT for table `object`
--
ALTER TABLE `object`
  MODIFY `ID` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=13;

--
-- AUTO_INCREMENT for table `tool`
--
ALTER TABLE `tool`
  MODIFY `ID` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=26;

--
-- AUTO_INCREMENT for table `user`
--
ALTER TABLE `user`
  MODIFY `ID` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=78;

--
-- Constraints for dumped tables
--

--
-- Constraints for table `comment`
--
ALTER TABLE `comment`
  ADD CONSTRAINT `comment_ibfk_1` FOREIGN KEY (`User_ID`) REFERENCES `user` (`ID`),
  ADD CONSTRAINT `comment_ibfk_2` FOREIGN KEY (`Tool_ID`) REFERENCES `tool` (`ID`);

--
-- Constraints for table `history`
--
ALTER TABLE `history`
  ADD CONSTRAINT `history_ibfk_1` FOREIGN KEY (`Whm1_ID`) REFERENCES `user` (`ID`),
  ADD CONSTRAINT `history_ibfk_2` FOREIGN KEY (`Object_ID`) REFERENCES `object` (`ID`),
  ADD CONSTRAINT `history_ibfk_3` FOREIGN KEY (`Storage_ID`) REFERENCES `object` (`ID`),

```

```

ADD CONSTRAINT `history_ibfk_4` FOREIGN KEY (`Tool_ID`) REFERENCES `tool` (`ID`),
ADD CONSTRAINT `history_ibfk_5` FOREIGN KEY (`Bldr_ID`) REFERENCES `user` (`ID`),
ADD CONSTRAINT `history_ibfk_6` FOREIGN KEY (`Whm2_ID`) REFERENCES `user` (`ID`);

--
-- Constraints for table `notification`
--
ALTER TABLE `notification`
  ADD CONSTRAINT `notification_ibfk_1` FOREIGN KEY (`User_ID`) REFERENCES `user` (`ID`),
  ADD CONSTRAINT `notification_ibfk_2` FOREIGN KEY (`Tool_ID`) REFERENCES `tool` (`ID`),
  ADD CONSTRAINT `notification_ibfk_3` FOREIGN KEY (`From_ID`) REFERENCES `user` (`ID`);

--
-- Constraints for table `tool`
--
ALTER TABLE `tool`
  ADD CONSTRAINT `tool_ibfk_1` FOREIGN KEY (`User_ID`) REFERENCES `user` (`ID`),
  ADD CONSTRAINT `tool_ibfk_2` FOREIGN KEY (`Object_ID`) REFERENCES `object` (`ID`);
COMMIT;

/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;

```

Додаток 3

Код сторінки “Інструменти”

```

<?php
require_once('../php/check_login.php');
session_start();
?>

<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="stylesheet" href="../css/tools.css">
  <link rel="icon" type="image/x-icon" href="../images/favicon4.png">
  <style>
    /* Ваші стилі тут */
  </style>
  <title>Інструменти</title>
</head>

<body>
  <div class="wrapper0">
    <div id="left-menu">
      <?php require_once('../php/check_login.php'); ?>
      <a href="tools.php" class="act">Інструменти</a>
      <a href="objects.php">Об'єкти</a>
      <a href="users.php">Користувачі</a>
      <a href="notifications.php">Повідомлення</a>

      <div id="exit">
        <?php
        echo "<div class='userinfo'>";
        echo "Login: ";
        echo "<p>" . $_SESSION['login'] . "</p>";
        echo "ID: ";

```

```

        echo "<p>" . $_SESSION['id'] . "</p>";
        echo "Type: ";
        echo "<p>" . $_SESSION['type'] . "</p>";
        echo "</div>";
        ?>
        <a href="../php/logout.php">Exit</a>
    </div>
</div>

<div class="wrapper">
    <div>
        <p>Инструменти</p>

        <div>
            <form class="search" method="get" action="tools.php" name="search-form">
                <div class="div1">
                    <input type="text" name="search" placeholder="Search" value="<?php echo
isset($_GET['search']) ? $_GET['search'] : "; ?>">
                    <button type="submit">Filter</button>
                    <button type="button" onclick="resetFilters()">Reset</button>
                    <?php
                    if ($_SESSION['type'] == "admin") {
                        echo "<a href='addtool.php' class='a_addtool'>+Add tool</a>";
                    }
                    ?>
                </div>

                <div class="cf">
                    <input type="text" name="filter_id" placeholder="ID" value="<?php echo
isset($_GET['filter_id']) ? $_GET['filter_id'] : "; ?>">
                    <input type="text" name="filter_user_id" placeholder="User ID" value="<?php echo
isset($_GET['filter_user_id']) ? $_GET['filter_user_id'] : "; ?>">
                    <input type="text" name="filter_object_id" placeholder="Object ID" value="<?php echo
isset($_GET['filter_object_id']) ? $_GET['filter_object_id'] : "; ?>">
                    <input type="text" name="filter_name" placeholder="Name" value="<?php echo
isset($_GET['filter_name']) ? $_GET['filter_name'] : "; ?>">
                    <input type="text" name="filter_type" placeholder="Type" value="<?php echo
isset($_GET['filter_type']) ? $_GET['filter_type'] : "; ?>">
                    <input type="text" name="filter_serial_number" placeholder="Serial Number"
value="<?php echo isset($_GET['filter_serial_number']) ? $_GET['filter_serial_number'] : "; ?>">
                    <input type="text" name="filter_status" placeholder="Status" value="<?php echo
isset($_GET['filter_status']) ? $_GET['filter_status'] : "; ?>">
                    <input type="text" name="filter_price" placeholder="Price" value="<?php echo
isset($_GET['filter_price']) ? $_GET['filter_price'] : "; ?>">
                    <input type="text" name="filter_details" placeholder="Details" value="<?php echo
isset($_GET['filter_details']) ? $_GET['filter_details'] : "; ?>">
                </div>

            </form>
        </div>

        <div class="container">
            <div class="item">ID</div>
            <div class="item">User_ID</div>
            <div class="item">Object_ID</div>
            <div class="item">Name</div>
            <div class="item">Type</div>
            <div class="item">SerialNumber</div>
            <div class="item">Status</div>

```

```
<div class="item">Price</div>
<div class="item">Details</div>
</div>
```

```
<?php
include('../php/connect_u.php');

$limit = 18;
$page = isset($_GET['page']) ? $_GET['page'] : 1;
$offset = ($page - 1) * $limit;

$search = isset($_GET['search']) ? $_GET['search'] : "";

$sql = "SELECT * FROM tool WHERE 1=1";
$countSql = "SELECT COUNT(*) as total FROM tool WHERE 1=1";
$params_types = "";
$params = array();

if (!empty($_GET['filter_id'])) {
    $sql .= " AND ID = ?";
    $countSql .= " AND ID = ?";
    $params_types .= "i";
    $params[] = $_GET['filter_id'];
}

if (!empty($_GET['filter_user_id'])) {
    $sql .= " AND User_ID = ?";
    $countSql .= " AND User_ID = ?";
    $params_types .= "i";
    $params[] = $_GET['filter_user_id'];
}

if (!empty($_GET['filter_object_id'])) {
    $sql .= " AND Object_ID = ?";
    $countSql .= " AND Object_ID = ?";
    $params_types .= "i";
    $params[] = $_GET['filter_object_id'];
}

if (!empty($_GET['filter_name'])) {
    $sql .= " AND Name LIKE ?";
    $countSql .= " AND Name LIKE ?";
    $params_types .= "s";
    $params[] = "%" . $_GET['filter_name'] . "%";
}

if (!empty($_GET['filter_type'])) {
    $sql .= " AND Type = ?";
    $countSql .= " AND Type = ?";
    $params_types .= "s";
    $params[] = $_GET['filter_type'];
}

if (!empty($_GET['filter_serial_number'])) {
    $sql .= " AND SerialNumber LIKE ?";
    $countSql .= " AND SerialNumber LIKE ?";
    $params_types .= "s";
    $params[] = "%" . $_GET['filter_serial_number'] . "%";
}
```

```

        if (!empty($_GET['filter_status'])) {
            $sql .= " AND Status = ?";
            $countSql .= " AND Status = ?";
            $param_types .= "s";
            $params[] = $_GET['filter_status'];
        }

        if (!empty($_GET['filter_price'])) {
            $sql .= " AND Price = ?";
            $countSql .= " AND Price = ?";
            $param_types .= "d";
            $params[] = $_GET['filter_price'];
        }

        if (!empty($search)) {
            $sql .= " AND (ID LIKE ? OR User_ID LIKE ? OR Object_ID LIKE ? OR Name LIKE ? OR
Type LIKE ? OR SerialNumber LIKE ? OR Status LIKE ? OR Price LIKE ? OR Details LIKE ?)";
            $countSql .= " AND (ID LIKE ? OR User_ID LIKE ? OR Object_ID LIKE ? OR Name LIKE ?
OR Type LIKE ? OR SerialNumber LIKE ? OR Status LIKE ? OR Price LIKE ? OR Details LIKE ?)";
            $param_types .= "ssssssss";
            $params = array_merge($params, array_fill(0, 9, "%$search%"));
        }

        $stmt = $conn->prepare($countSql);

        if (!empty($params)) {
            $stmt->bind_param($param_types, ...$params);
        }

        $stmt->execute();
        $result = $stmt->get_result();
        $total_records = $result->fetch_assoc()['total'];

        $total_pages = ceil($total_records / $limit);

        $sql .= " LIMIT $limit OFFSET $offset";

        $stmt = $conn->prepare($sql);

        if (!empty($params)) {
            $stmt->bind_param($param_types, ...$params);
        }

        $stmt->execute();
        $result = $stmt->get_result();

        if (mysqli_num_rows($result) > 0) {
            while ($row = $result->fetch_assoc()) {
                echo "<div class='container'>";
                echo "<div class='item'><a href='/pages/tool_info.php?tool_ID=" . $row['ID'] . ">" .
$row["ID"] . "</a></div>";
                echo "<div class='item'><a href='/pages/tool_info.php?tool_ID=" . $row['ID'] . ">" .
$row["User_ID"] . "</a></div>";
                echo "<div class='item'><a href='/pages/tool_info.php?tool_ID=" . $row['ID'] . ">" .
$row["Object_ID"] . "</a></div>";
                echo "<div class='item'><a href='/pages/tool_info.php?tool_ID=" . $row['ID'] . ">" .
$row["Name"] . "</a></div>";
                echo "<div class='item'><a href='/pages/tool_info.php?tool_ID=" . $row['ID'] . ">" .
$row["Type"] . "</a></div>";
            }
        }

```

```

                echo "<div class='item'><a href='/pages/tool_info.php?tool_ID=" . $row['ID'] . "'>" .
$row["SerialNumber"] . "</a></div>";
                echo "<div class='item'><a href='/pages/tool_info.php?tool_ID=" . $row['ID'] . "'>" .
$row["Status"] . "</a></div>";
                echo "<div class='item'><a href='/pages/tool_info.php?tool_ID=" . $row['ID'] . "'>" .
$row["Price"] . "</a></div>";
                echo "<div class='item'><a href='/pages/tool_info.php?tool_ID=" . $row['ID'] . "'>" .
$row["Details"] . "</a></div>";
                echo "</div>";
            }

            echo "<p class='num'>";
            $start_page = max(1, $page - 5);
            $end_page = min($total_pages, $page + 5);

            $base_url = 'tools.php?' . http_build_query(array_diff_key($_GET, ['page' => "]));

            if ($page > 1) {
                echo "<a class='pages' href='{ $base_url }&page=1'>First</a> ";
                echo "<a class='pages' href='{ $base_url }&page=" . ($page - 1) . "'>&larr;</a> ";
            }

            for ($i = $start_page; $i <= $end_page; $i++) {
                echo "<a class='pages' href='{ $base_url }&page=" . $i . "'>" . $i . "</a> ";
            }

            if ($page < $total_pages) {
                echo "<a class='pages' href='{ $base_url }&page=" . ($page + 1) . "'>&rarr;</a> ";
                echo "<a class='pages' href='{ $base_url }&page=" . $total_pages . "'>Last</a> ";
            }

            echo "</p>";
        } else {
            echo "0 результатів";
        }

        $conn->close();
    ?>
</div>
</div>
</div>

<script>
function resetFilters() {
    document.getElementsByName('filter_id')[0].value = "";
    document.getElementsByName('filter_user_id')[0].value = "";
    document.getElementsByName('filter_object_id')[0].value = "";
    document.getElementsByName('filter_name')[0].value = "";
    document.getElementsByName('filter_type')[0].value = "";
    document.getElementsByName('filter_serial_number')[0].value = "";
    document.getElementsByName('filter_status')[0].value = "";
    document.getElementsByName('filter_price')[0].value = "";
    document.getElementsByName('search')[0].value = "";
    document.forms['search-form'].submit();
}
</script>
</body>

</html>

```

Додаток 4

Код сторінки “Повідомлення”

```
<?php
require_once('../php/check_login.php');
session_start();
?>

<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="../css/notifications.css">
    <link rel="icon" type="image/x-icon" href="../images/favicon4.png">
    <style>
        /* Your styles here */
    </style>
    <title>Notifications</title>
</head>

<body>
    <div class="wrapper0">
        <div id="left-menu">
            <?php require_once('../php/check_login.php'); ?>
            <a href="tools.php">Інструменти</a>
            <a href="objects.php">Об'єкти</a>
            <a href="users.php">Користувачі</a>
            <a href="notifications.php" class="act">Повідомлення</a>

            <div id="exit">
                <?php
                echo "<div class='userinfo'>";
                echo "Login: ";
                echo "<p>" . $_SESSION['login'] . "</p>";
                echo "ID: ";
                echo "<p>" . $_SESSION['id'] . "</p>";
                echo "Type: ";
                echo "<p>" . $_SESSION['type'] . "</p>";
                echo "</div>";
                ?>
                <a href="../php/logout.php">Exit</a>
            </div>
        </div>

        <div class="wrapper">
            <div>
                <p>Повідомлення</p>

                <div>
                    <form class="search" method="get" action="notifications.php" name="search-form">
                        <div class="div1">
                            <input type="text" name="search" placeholder="Search" value="<?php echo isset($_GET['search'])
? $_GET['search'] : ''; ?>">
                            <button type="submit">Filter</button>
                            <button type="button" onclick="resetFilters()">Reset</button>
                        </div>

                        <div class="cf">
```

```

        <input type="text" name="filter_id" placeholder="ID" value="<?php echo isset($_GET['filter_id'])
? $_GET['filter_id'] : ""; ?>">
        <input type="text" name="filter_from_id" placeholder="From_ID" value="<?php echo
isset($_GET['filter_from_id']) ? $_GET['filter_from_id'] : ""; ?>">
        <input type="text" name="#" placeholder=" " value=" " disabled>
        <input type="text" name="filter_tool_id" placeholder="Tool_ID" value="<?php echo
isset($_GET['filter_tool_id']) ? $_GET['filter_tool_id'] : ""; ?>">
        <input type="text" name="filter_action" placeholder="Action" value="<?php echo
isset($_GET['filter_action']) ? $_GET['filter_action'] : ""; ?>">
        <input type="text" name="filter_status" placeholder="Status" value="<?php echo
isset($_GET['filter_status']) ? $_GET['filter_status'] : ""; ?>">
        <input type="text" name="filter_datetime" placeholder="Datetime" value="<?php echo
isset($_GET['filter_datetime']) ? $_GET['filter_datetime'] : ""; ?>">
    </div>
</form>
</div>

```

```

<div class="container">
    <div class="item">ID</div>
    <div class="item">From_ID</div>
    <div class="item">User_ID</div>
    <div class="item">Tool_ID</div>
    <div class="item">Action</div>
    <div class="item">Status</div>
    <div class="item">Datetime</div>
</div>

```

```

<?php
include('../php/connect_u.php');

```

```

$limit = 18;
$page = isset($_GET['page']) ? $_GET['page'] : 1;
$offset = ($page - 1) * $limit;

```

```

$sql = "SELECT * FROM notification WHERE User_ID = ?";
$countSql = "SELECT COUNT(*) as total FROM notification WHERE User_ID = ?";
$params_types = "i";
$params = array($_SESSION['id']); // Use the session id for filtering User_ID

```

```

// Handling filters
$filterParams = array();
if (!empty($_GET['filter_id'])) {
    $sql .= " AND ID = ?";
    $countSql .= " AND ID = ?";
    $params_types .= "i";
    $filterParams[] = $_GET['filter_id'];
}

if (!empty($_GET['filter_from_id'])) {
    $sql .= " AND From_ID = ?";
    $countSql .= " AND From_ID = ?";
    $params_types .= "i";
    $filterParams[] = $_GET['filter_from_id'];
}

if (!empty($_GET['filter_tool_id'])) {
    $sql .= " AND Tool_ID = ?";
    $countSql .= " AND Tool_ID = ?";
    $params_types .= "i";
    $filterParams[] = $_GET['filter_tool_id'];
}

```

```

if (!empty($_GET['filter_action'])) {
    $sql .= " AND Action = ?";
    $countSql .= " AND Action = ?";
    $param_types .= "s";
    $filterParams[] = $_GET['filter_action'];
}

if (!empty($_GET['filter_status'])) {
    $sql .= " AND Status = ?";
    $countSql .= " AND Status = ?";
    $param_types .= "s";
    $filterParams[] = $_GET['filter_status'];
}

if (!empty($_GET['filter_datetime'])) {
    $sql .= " AND Datetime LIKE ?";
    $countSql .= " AND Datetime LIKE ?";
    $param_types .= "s";
    $filterParams[] = "%" . $_GET['filter_datetime'] . "%";
}

// Merge filter parameters
$params = array_merge($params, $filterParams);

// Handling search
$searchParams = array();
$search = isset($_GET['search']) ? $_GET['search'] : "";
if (!empty($search)) {
    $sql .= " AND (ID LIKE ? OR From_ID LIKE ? OR Tool_ID LIKE ? OR Action LIKE ? OR Status
LIKE ? OR Datetime LIKE ?)";
    $countSql .= " AND (ID LIKE ? OR From_ID LIKE ? OR Tool_ID LIKE ? OR Action LIKE ? OR Status
LIKE ? OR Datetime LIKE ?)";
    $param_types .= "ssssss";
    $searchParams = array_merge(array_fill(0, 6, "%$search%"));
}

// Merge search parameters
$params = array_merge($params, $searchParams);

$stmt = $conn->prepare($countSql);
$stmt->bind_param($param_types, ...$params);
$stmt->execute();
$result = $stmt->get_result();
$total_records = $result->fetch_assoc()['total'];

$total_pages = ceil($total_records / $limit);

$sql .= " LIMIT $limit OFFSET $offset";

$stmt = $conn->prepare($sql);
$stmt->bind_param($param_types, ...$params);
$stmt->execute();
$result = $stmt->get_result();

if (mysqli_num_rows($result) > 0) {
    while ($row = $result->fetch_assoc()) {
        echo "<div class='container'>";
        echo " <div class='item'><a href='/pages/tool_info.php?tool_ID=" . $row["Tool_ID"]. ">" .
$row["ID"] . "</a></div>";
    }
}

```

```

        echo "<div class='item'><a href='/pages/tool_info.php?tool_ID=" . $row["Tool_ID"]. ">" .
$row["From_ID"] . "</a></div>";
        echo "<div class='item'><a href='/pages/tool_info.php?tool_ID=" . $row["Tool_ID"]. ">" .
$row["User_ID"] . "</a></div>";
        echo "<div class='item'><a href='/pages/tool_info.php?tool_ID=" . $row["Tool_ID"]. ">" .
$row["Tool_ID"] . "</a></div>";
        echo "<div class='item'><a href='/pages/tool_info.php?tool_ID=" . $row["Tool_ID"]. ">" .
$row["Action"] . "</a></div>";
        echo "<div class='item'><a href='/pages/tool_info.php?tool_ID=" . $row["Tool_ID"]. ">" .
$row["Status"] . "</a></div>";
        echo "<div class='item'><a href='/pages/tool_info.php?tool_ID=" . $row["Tool_ID"]. ">" .
$row["Datetime"] . "</a></div>";
        echo "</div>";
    }

    echo "<p class='num'>";
    $start_page = max(1, $page - 5);
    $end_page = min($total_pages, $page + 5);

    $base_url = 'notifications.php?' . http_build_query(array_merge($_GET, array('page' => ",
'filter_User_ID' => $_SESSION['id'])));

    if ($page > 1) {
        echo "<a class='pages' href='{ $base_url }&page=1'>First</a> ";
        echo "<a class='pages' href='{ $base_url }&page=" . ($page - 1) . "'>&larr;</a> ";
    }

    for ($i = $start_page; $i <= $end_page; $i++) {
        echo "<a class='pages' href='{ $base_url }&page=" . $i . "'>" . $i . "</a> ";
    }

    if ($page < $total_pages) {
        echo "<a class='pages' href='{ $base_url }&page=" . ($page + 1) . "'>&rarr;</a> ";
        echo "<a class='pages' href='{ $base_url }&page=" . $total_pages . "'>Last</a> ";
    }

    echo "</p>";
} else {
    echo "0 результатів";
}

$conn->close();
?>
</div>
</div>
</div>
<script>
function resetFilters() {
    document.getElementById('filter_id')[0].value = "";
    document.getElementById('filter_from_id')[0].value = "";
    document.getElementById('filter_tool_id')[0].value = "";
    document.getElementById('filter_action')[0].value = "";
    document.getElementById('filter_status')[0].value = "";
    document.getElementById('filter_datetime')[0].value = "";
    document.getElementById('search')[0].value = "";
    document.forms['search-form'].submit();
}
</script>
</body>
</html>

```

Додаток 5

Код сторінки “Об’єкти”

```
<?php
require_once('../php/check_login.php');
session_start();
?>

<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="../css/objects.css">
    <link rel="icon" type="image/x-icon" href="../images/favicon4.png">
    <style>
        /* Ваші стилі тут */
    </style>
    <title>Об’єкти</title>
</head>

<body>
    <div class="wrapper0">
        <div id="left-menu">
            <?php require_once('../php/check_login.php'); ?>

            <a href="tools.php">Інструменти</a>
            <a href="objects.php" class="act">Об’єкти</a>
            <a href="users.php">Користувачі</a>
            <a href="notifications.php">Повідомлення</a>

            <div id="exit">
                <?php
                echo "<div class='userinfo'>";
                echo "Login: ";
                echo "<p>" . $_SESSION['login'] . "</p>";
                echo "ID: ";
                echo "<p>" . $_SESSION['id'] . "</p>";
                echo "Type: ";
                echo "<p>" . $_SESSION['type'] . "</p>";
                echo "</div>";
                ?>
                <a href="../php/logout.php">Exit</a>
            </div>
        </div>

        <div class="wrapper">
            <div>
                <p>Об’єкти</p>

                <div >
                    <form class="search" method="get" action="objects.php" name="search-form">
                        <div class="div1">
                            <input type="text" name="search" placeholder="Пошук" value="<?php echo
isset($_GET['search']) ? $_GET['search'] : ''; ?>">
                            <button type="submit">Filter</button>
                            <button type="button" onclick="resetFilters()">Reset</button>
                            <!-- Додайте інші фільтри за необхідністю -->
                        </div>
                    </form>
                </div>
            </div>
        </div>
    </div>
</body>
</html>
```

```

        <div class="cf">
            <input type="text" name="filter_id" placeholder="ID" value="<?php echo isset($_GET['filter_id'])
? $_GET['filter_id'] : ""; ?>">
            <input type="text" name="filter_name" placeholder="Назва" value="<?php echo
isset($_GET['filter_name']) ? $_GET['filter_name'] : ""; ?>">
            <input type="text" name="filter_details" placeholder="Деталі" value="<?php echo
isset($_GET['filter_details']) ? $_GET['filter_details'] : ""; ?>">
            <input type="text" name="filter_link" placeholder="Посилання" value="<?php echo
isset($_GET['filter_link']) ? $_GET['filter_link'] : ""; ?>">
        </div>
    </form>
</div>

```

```

<div class="container">
    <div class="item">ID</div>
    <div class="item">Назва</div>
    <div class="item">Деталі</div>
    <div class="item">Посилання</div>
</div>

```

```

<?php
include('../php/connect_u.php');

```

```

$limit = 18;
$page = isset($_GET['page']) ? $_GET['page'] : 1;
$offset = ($page - 1) * $limit;

```

```

$search = isset($_GET['search']) ? $_GET['search'] : "";

```

```

$sql = "SELECT * FROM object WHERE 1=1";
$countSql = "SELECT COUNT(*) as total FROM object WHERE 1=1";
$params_types = "";
$params = array();

```

```

if (!empty($_GET['filter_id'])) {
    $sql .= " AND ID = ?";
    $countSql .= " AND ID = ?";
    $params_types .= "i";
    $params[] = $_GET['filter_id'];
}

```

```

if (!empty($_GET['filter_name'])) {
    $sql .= " AND Name LIKE ?";
    $countSql .= " AND Name LIKE ?";
    $params_types .= "s";
    $params[] = "%" . $_GET['filter_name'] . "%";
}

```

```

if (!empty($_GET['filter_details'])) {
    $sql .= " AND Details LIKE ?";
    $countSql .= " AND Details LIKE ?";
    $params_types .= "s";
    $params[] = "%" . $_GET['filter_details'] . "%";
}

```

```

if (!empty($_GET['filter_link'])) {
    $sql .= " AND Link LIKE ?";
    $countSql .= " AND Link LIKE ?";
    $params_types .= "s";
    $params[] = "%" . $_GET['filter_link'] . "%";
}

```

```

if (!empty($search)) {
    $sql .= " AND (ID LIKE ? OR Name LIKE ? OR Details LIKE ? OR Link LIKE ?)";
    $countSql .= " AND (ID LIKE ? OR Name LIKE ? OR Details LIKE ? OR Link LIKE ?)";
    $param_types .= "ssss";
    $params = array_merge($params, array_fill(0, 4, "%$search%"));
}

$stmt = $conn->prepare($countSql);

if (!empty($params)) {
    $stmt->bind_param($param_types, ...$params);
}

$stmt->execute();
$result = $stmt->get_result();
$total_records = $result->fetch_assoc()['total'];

$total_pages = ceil($total_records / $limit);

$sql .= " LIMIT $limit OFFSET $offset";

$stmt = $conn->prepare($sql);

if (!empty($params)) {
    $stmt->bind_param($param_types, ...$params);
}

$stmt->execute();
$result = $stmt->get_result();

if (mysqli_num_rows($result) > 0) {
    while ($row = $result->fetch_assoc()) {
        echo "<div class='container'>";
        echo "<div class='item'>" . $row["ID"] . "</div>";
        echo "<div class='item'>" . $row["Name"] . "</div>";
        echo "<div class='item'>" . $row["Details"] . "</div>";
        echo "<div class='item'><a href='" . $row["Link"] . "' target='_blank'>" . $row["Link"] . "</a></div>";

        echo "</div>";
    }

    echo "<p class='num'>";
    $start_page = max(1, $page - 5);
    $end_page = min($total_pages, $page + 5);

    $base_url = 'objects.php?' . http_build_query(array_diff_key($_GET, ['page' => "]));

    if ($page > 1) {
        echo "<a class='pages' href='{ $base_url }&page=1'>First</a> ";
        echo "<a class='pages' href='{ $base_url }&page=" . ($page - 1) . "'>&larr;</a> ";
    }

    for ($i = $start_page; $i <= $end_page; $i++) {
        echo "<a class='pages' href='{ $base_url }&page=" . $i . "'>" . $i . "</a> ";
    }

    if ($page < $total_pages) {
        echo "<a class='pages' href='{ $base_url }&page=" . ($page + 1) . "'>&rarr;</a> ";
        echo "<a class='pages' href='{ $base_url }&page=" . $total_pages . "'>Last</a> ";
    }
}

```

```

        echo "</p>";
    } else {
        echo "0 результатів";
    }

    $conn->close();
?>
</div>
</div>
</div>

<script>
function resetFilters() {
    document.getElementById('filter_id')[0].value = "";
    document.getElementById('filter_name')[0].value = "";
    document.getElementById('filter_details')[0].value = "";
    document.getElementById('filter_link')[0].value = "";
    document.getElementById('search')[0].value = "";
    document.forms['search-form'].submit();
}
</script>
</body>

</html>

```

Додаток 6

```
<?php
require_once('../php/check_login.php');
require_once('notifstatus.php');

session_start();
?>

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="../css/tool_info.css">
    <link rel="icon" type="image/x-icon" href="../images/favicon4.png">
    <title>Tool Details</title>
</head>
<body>
<div class="wrapper0">
    <div id="left-menu">
        <a href="tools.php">Інструменти</a>
        <a href="objects.php">Об'єкти</a>
        <a href="users.php">Користувачі</a>
        <a href="notifications.php">Повідомлення</a>
    <div id="exit">
        <?php // Session information
        echo "<div class='userinfo'>";
        echo "Login: ";
        echo "<p>" . $_SESSION['login'] . "</p>";
        echo "User type: ";
        echo "<p>" . $_SESSION['type'] . "</p>";
        echo "</div>";
    <?php
    ?>
    ?>
```

```

        <a href="../php/logout.php">Exit</a>
    </div>
</div>

<?php
// Підключення до бд
require_once('../php/connect_a.php');

// Отримання параметру tool_ID з URL
$tool_id = $_GET['tool_ID'];

// Отримання даних з бази даних для конкретного інструменту
$tool_query = $conn->prepare("SELECT * FROM tool WHERE ID = ?");
$tool_query->bind_param("i", $tool_id);

if ($tool_query->execute()) {
    $tool_result = $tool_query->get_result();

    if ($tool_result->num_rows > 0) {
        // Інструмент знайдено, вивести дані
        $tool_data = $tool_result->fetch_assoc();
        // ... (Display tool details as before)
        ?>
        <div class="wrapper">

            <div class="cmnts">
                <?php
                    include 'comments.php';
                ?>
            </div>

            <div class="wrapper0">
                <h2>Tool Details</h2>
                <p>ID: <?php echo $tool_data['ID']; ?></p>
                <p>User ID: <?php echo $tool_data['User_ID']; ?></p>
                <p>Object ID: <?php echo $tool_data['Object_ID']; ?></p>
                <p>Name: <?php echo $tool_data['Name']; ?></p>
                <p>Type: <?php echo $tool_data['Type']; ?></p>
                <p>Serial Number: <?php echo $tool_data['SerialNumber']; ?></p>
                <p>Status: <?php echo $tool_data['Status']; ?></p>
                <p>Price: <?php echo $tool_data['Price']; ?></p>
                <p>Image: <?php echo $tool_data['Image']; ?></p>
                <p>Details: <?php echo $tool_data['Details']; ?></p>
            </div>

            <?php

            include 'history.php';
            ?>
        </div>
        <?php
            include 'request_buttons.php';
            ?>

        <?php
    } else {
        echo "<p>No results found for tool ID: $tool_id</p>";
    }
} else {

```

```

        echo "<p>Error executing query: " . $conn->error . "</p>";
    }

    $tool_query->close();
    $conn->close();
?>

</body>
</html>

```

Додаток 7

Код сторінки “Користувачі”

```

<?php
require_once('../php/check_login.php');
session_start();
?>

<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="../css/users.css">
    <link rel="icon" type="image/x-icon" href="../images/favicon4.png">
    <style>
        /* Your styles here */
    </style>
    <title>Table user</title>
</head>

<body>
    <div class="wrapper0">
        <div id="left-menu">
            <a href="tools.php">Інструменти</a>
            <a href="objects.php">Об'єкти</a>
            <a href="users.php" class="act">Користувачі</a>
            <a href="notifications.php">Повідомлення</a>

            <div id="exit">
                <?php
                echo "<div class='userinfo'>";
                echo "Login: ";
                echo "<p>" . $_SESSION['login'] . "</p>";
                echo "ID: ";
                echo "<p>" . $_SESSION['id'] . "</p>";
                echo "User type: ";
                echo "<p>" . $_SESSION['type'] . "</p>";
                echo "</div>";
                ?>
                <a href="../php/logout.php">Exit</a>
            </div>
        </div>

        <div class="wrapper">
            <div>
                <p>Table user</p>

                <form class="search" method="get" action="users.php" name="search-form">

```

```

<div class="div1">
    <input style="margin-right: 10px;" type="text" name="search" placeholder="Search"
        value="<?php echo isset($_GET['search']) ? $_GET['search'] : ''; ?>">
    <button type="submit">Filter</button>
    <button type="button" onclick="resetFilters()">Reset</button>
</div>
<div class="cf">
    <input type="text" name="filter_id" placeholder="ID"
        value="<?php echo isset($_GET['filter_id']) ? $_GET['filter_id'] : ''; ?>">
    <input type="text" name="filter_name" placeholder="Name"
        value="<?php echo isset($_GET['filter_name']) ? $_GET['filter_name'] : ''; ?>">
    <input type="text" name="filter_lastname" placeholder="Lastname"
        value="<?php echo isset($_GET['filter_lastname']) ? $_GET['filter_lastname'] : ''; ?>">
    <input type="text" name="filter_email" placeholder="E-mail"
        value="<?php echo isset($_GET['filter_email']) ? $_GET['filter_email'] : ''; ?>">
    <input type="text" name="filter_type" placeholder="Type"
        value="<?php echo isset($_GET['filter_type']) ? $_GET['filter_type'] : ''; ?>">
</div>
</form>

<div class="container">
    <div class="item">ID</div>
    <div class="item">Name</div>
    <div class="item">Lastname</div>
    <div class="item">E-mail</div>
    <div class="item">Type</div>
</div>

<?php
include('../php/connect_u.php');

$limit = 18;
$page = isset($_GET['page']) ? $_GET['page'] : 1;
$offset = ($page - 1) * $limit;

$filter_id = isset($_GET['filter_id']) ? $_GET['filter_id'] : '';
$filter_name = isset($_GET['filter_name']) ? $_GET['filter_name'] : '';
$filter_lastname = isset($_GET['filter_lastname']) ? $_GET['filter_lastname'] : '';
$filter_email = isset($_GET['filter_email']) ? $_GET['filter_email'] : '';
$filter_type = isset($_GET['filter_type']) ? $_GET['filter_type'] : '';
$search = isset($_GET['search']) ? $_GET['search'] : '';

$sql = "SELECT * FROM user WHERE 1=1";
$params_types = "";
$params = array();

if (!empty($filter_id)) {
    $sql .= " AND ID LIKE ?";
    $params_types .= "s";
    $params[] = "%$filter_id%";
}
if (!empty($filter_name)) {
    $sql .= " AND Name LIKE ?";
    $params_types .= "s";
    $params[] = "%$filter_name%";
}
if (!empty($filter_lastname)) {
    $sql .= " AND Lastname LIKE ?";
    $params_types .= "s";
    $params[] = "%$filter_lastname%";
}

```

```

if (!empty($filter_email)) {
    $sql .= " AND E_mail LIKE ?";
    $param_types .= "s";
    $params[] = "%$filter_email%";
}
if (!empty($filter_type)) {
    $sql .= " AND Type LIKE ?";
    $param_types .= "s";
    $params[] = "%$filter_type%";
}
if (!empty($search)) {
    $sql .= " AND (ID LIKE ? OR Name LIKE ? OR Lastname LIKE ? OR E_mail LIKE ? OR Type LIKE
?)" ;
    $param_types .= "sssss";
    $params = array_merge($params, array_fill(0, 5, "%$search%"));
}

$sql .= " LIMIT $limit OFFSET $offset";

$stmt = $conn->prepare($sql);

if (!empty($param_types)) {
    $bind_params = array_merge([$param_types], $params);
    if (is_array($bind_params)) {
        call_user_func_array(array($stmt, 'bind_param'), $bind_params);
    }
}

$stmt->execute();
$result = $stmt->get_result();

if (mysqli_num_rows($result) > 0) {
    while ($row = $result->fetch_assoc()) {
        echo "<div class='container'>";
        echo "<div class='item'>" . $row["ID"] . "</div>";
        echo "<div class='item'>" . $row["Name"] . "</div>";
        echo "<div class='item'>" . $row["Lastname"] . "</div>";
        echo "<div class='item'>" . $row["E_mail"] . "</div>";
        echo "<div class='item'>" . $row["Type"] . "</div>";
        echo "</div>";
    }

    $total_records_sql = "SELECT COUNT(*) FROM user WHERE 1=1";
    $param_types = "";
    $params = array();

    if (!empty($filter_id)) {
        $total_records_sql .= " AND ID LIKE ?";
        $param_types .= "s";
        $params[] = "%$filter_id%";
    }
    if (!empty($filter_name)) {
        $total_records_sql .= " AND Name LIKE ?";
        $param_types .= "s";
        $params[] = "%$filter_name%";
    }
    if (!empty($filter_lastname)) {
        $total_records_sql .= " AND Lastname LIKE ?";
        $param_types .= "s";
        $params[] = "%$filter_lastname%";
    }
}

```

```

if (!empty($filter_email)) {
    $total_records_sql .= " AND E_mail LIKE ?";
    $param_types .= "s";
    $params[] = "%$filter_email%";
}
if (!empty($filter_type)) {
    $total_records_sql .= " AND Type LIKE ?";
    $param_types .= "s";
    $params[] = "%$filter_type%";
}
if (!empty($search)) {
    $total_records_sql .= " AND (ID LIKE ? OR Name LIKE ? OR Lastname LIKE ? OR E_mail LIKE ?
OR Type LIKE ?)";
    $param_types .= "sssss";
    $params = array_merge($params, array_fill(0, 5, "%$search%"));
}

$total_records_stmt = $conn->prepare($total_records_sql);

if (!empty($param_types)) {
    $bind_params = array_merge([$param_types], $params);
    if (is_array($bind_params)) {
        call_user_func_array(array($total_records_stmt, 'bind_param'), $bind_params);
    }
}

$total_records_stmt->execute();
$total_records_result = $total_records_stmt->get_result();
$total_records = $total_records_result->fetch_array()[0];

$total_pages = ceil($total_records / $limit);

echo "<p class='num'>";
$start_page = max(1, $page - 5);
$end_page = min($total_pages, $page + 5);

$base_url = 'users.php?' . http_build_query(array_diff_key($_GET, ['page' => "]));

if ($page > 1) {
    echo "<a class='pages' href='{ $base_url }&page=1'>First</a> ";
    echo "<a class='pages' href='{ $base_url }&page=" . ($page - 1) . "'>&larr;</a> ";
}

for ($i = $start_page; $i <= $end_page; $i++) {
    echo "<a class='pages' href='{ $base_url }&page=" . $i . "'>" . $i . "</a> ";
}

if ($page < $total_pages) {
    echo "<a class='pages' href='{ $base_url }&page=" . ($page + 1) . "'>&rarr;</a> ";
    echo "<a class='pages' href='{ $base_url }&page=" . $total_pages . "'>Last</a> ";
}

echo "</p>";
} else {
    echo "0 результатов";
}

$conn->close();
?>
</div>
</div>

```

```
</div>
```

```
<script>
```

```
function resetFilters() {  
    document.getElementsByName('filter_id')[0].value = "";  
    document.getElementsByName('filter_name')[0].value = "";  
    document.getElementsByName('filter_lastname')[0].value = "";  
    document.getElementsByName('filter_email')[0].value = "";  
    document.getElementsByName('filter_type')[0].value = "";  
    document.getElementsByName('search')[0].value = "";  
    document.forms['search-form'].submit();  
}
```

```
</script>
```

```
</body>
```

```
</html>
```