

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»**

Факультет інформатики та обчислювальної техніки _____
(повна назва інституту/факультету)

Кафедра автоматичного управління в технічних системах _____
(повна назва кафедри)

«На правах рукопису»
УДК 004.75 _____

«До захисту допущено»
Завідувач кафедри
_____ О.І. Ролик
(підпис) (ініціали,
прізвище)

“ ” _____ 2018р.

Магістерська дисертація

зі спеціальності (спеціалізації) 121 Інженерія програмного забезпечення
(код і назва спеціальності)

на тему: ”Система управління складським обліком

Виконав (-ла): студент (-ка) б курсу, групи ІТ-73мп
(шифр групи)

Чорноус Катерини Володимирівни _____
(прізвище, ім'я, по батькові) (підпис)

Науковий керівник к.т.н., доцент Букасов М.М. _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант _____
(назва розділу) (науковий ступінь, вчене звання, прізвище, ініціали) (підпис)

Рецензент _____
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій магістерській
дисертації немає запозичень з праць
інших авторів без відповідних посилань.

Студент _____
(підпис)

додатків, забезпечення зчитування штрих коду товару та додавання його до системи.

5. Перелік завдань, які потрібно розробити:

- розробити архітектурне рішення системи ;
- розробити клієнтську та серверну частину системи;
- розробити компонент зчитування штрих коду;
- розробити компонент пошуку по всій системі;
- розробити графічний інтерфейс для двох додатків для відображення.

6. Орієнтовний перелік ілюстративного (графічного) матеріалу: ER діаграма бази даних, UML діаграма станів системи, Структура схема системи, UML діаграма процесу доставки нового товару, UML діаграма процесу переміщення товару, UML діаграма розгортання, Use Case діаграма варіантів використання мобільного додатку, USE Case діаграма варіантів використання системи з боку оператора та менеджера.

7. Дата видачі завдання: 28 жовтня 2018 рік.

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Строк виконання етапів роботи	Примітка
1	Ознайомлення із завданням	29.10.18 – 01.11.18	
2	Огляд предметної області та існуючих рішень	02.11.18 – 05.11.18	
3	Визначення вимог до системи	06.11.18 – 08.11.18	
4	Вибір технологій та реалізацій ПЗ	09.11.18 – 10.11.18	
5	Проектування та розробка архітектури системи	11.11.18 – 18.11.18	
6	Реалізація функціоналу системи	19.11.18 – 25.11.18	
7	Тестування системи	26.11.18 – 28.11.18	
8	Розроблення схем, ілюстрацій	29.11.18 – 30.11.18	
9	Оформлення пояснювальної записки	01.12.18 – 03.12.18	
10	Захист дипломного проекту	20.12.18	

Студент

(підпис)

Чорноус К.В.

(ініціали, прізвище)

Керівник проекту

(підпис)

Букасов М. М.

(ініціали, прізвище)

РЕФЕРАТ

Магістерська дисертація «Система управління складським обліком» складається з 101 аркуша пояснювальної записки, робота містить 36 рисунків, 39 таблиці, 8 креслеників та 14 бібліографічних посилань на використані джерела.

Актуальність теми магістерської дисертації зумовлена великою популярністю систем контролю та автоматизації підприємств.

Метою дипломного проекту є створення системи управління складським обліком. У дипломному проекті було спроектовано систему, що буде корисна як в великих там і малих підприємствах. Додаток призначений для обліку товару, контролю наявності вільних місць на складі, відстежування товару, вистежування прийняття товару в постачальника та відправки клієнту. Інтерфейс дозволяє користувачу взаємодіяти з програмним забезпеченням за допомогою візуального зображення та візуальних підказок. Під час розробки проекту застосувалися сучасні програмні продукти та технології.

Об'єктом дослідження є данні підприємства, які зберігаються на складі.

Наукова новизна роботи полягає в розробці системи, яка б дозволила автоматизувати будь-яке підприємство, та дати змогу віддалено керувати ним.

Результати розробки було впроваджено до компанії ТОВ «Наш зелений світ» з метою знизити витрати підприємства та покращити роботу. Впровадження результатів підтверджується актом про впровадження.

Перелік ключових слів: система, пошук, складське приміщення, автоматизація, сканування.

SUMMARY

Master's dissertation "The warehouse accounting management system" consists of 101 pages of the explanatory note, the work contains 36 drawings, 39 tables, 8 drawings and 14 bibliographic references on the used sources.

The relevance of the master's thesis topic is due to the great popularity of control and automation systems of enterprises.

The purpose of this diploma project is to create a warehouse management system. In the master's thesis project the system that was designed would be useful both in large and small enterprises. The application is intended for the goods accounting, availability control of spare places in the warehouse, tracking the goods, tracing the acceptance of the goods in the supplier and sending them to the client. The interface allows the user to interact with the software using a visual image and visual prompts. During development of the project, modern software products and technologies were used.

The research object is the data of enterprises that are stored in the warehouse.

The scientific novelty of the work is to develop a system that would allow the automation of any enterprise, and allow remote control of the NWM.

The results of the development were introduced to the company "Our Green World" Ltd. in order to reduce the costs of the company and improve the work. The implementation of the results is confirmed by the implementation act.

Keywords list: system, search, warehouse, automation, scanning.

Зміст

ВСТУП.....	8
1 ПОСТАНОВКА ЗАДАЧІ.....	10
1.1 Автоматизовані системи управління.....	11
1.2 Сутність ERP систем.....	14
1.3 Аналіз підприємства.....	21
1.4 Визначення необхідних функцій системи.....	23
1.5 Принцип єдиної бази.....	24
1.6 Урахування змін ринку.....	25
2 ОГЛЯ ІСНУЮЧИХ РІШЕНЬ.....	26
3 РОЗРОБКА СИСТЕМИ АВТОМАТИЗАЦІЇ.....	30
3.1 Формування вимог до системи.....	30
3.2 Логіко – структурна матриця.....	31
4 РЕАЛІЗАЦІЯ БІЗНЕС ЛОГІКИ.....	35
4.1 Описання алгоритму реалізації.....	35
4.2 Модульна структура системи.....	38
5 РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ.....	42
5.1 Реалізація клієнт серверної взаємодії.....	42
5.2 Десктопна частина системи.....	43
5.3 Проектування бази даних.....	49
5.4 Реалізація API.....	56
5.5 Реалізація програмного додатку.....	60
6 ТЕСТУВАННЯ СИСТЕМИ.....	68

6.1 Модульні тести.....	68
6.2 Інтеграційне тестування.....	70
6.3 Системне тестування.....	72
6.2 UI тести.....	73
7 СТАРТАП ПРОЕКТ.....	79
7.1 Опис ідеї стартап проекту.....	81
7.2 Технологічний аудит проекту.....	84
7.3 Аналіз ринкових можливостей запуску.....	84
7.4 Розроблення ринкової стратегії проекту.....	93
7.5 Розроблення маркетингової програми.....	95
7.6 Висновки до розділу.....	100
ВИСНОВК.....	101
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	102

ВСТУП

Тема даного дипломного проекту – Система управління Складським обліком. WCMS система є аббревіатурою Warehouse Complex Management System, що має в собі три програмних рішення. Перше програмне рішення має системну частину. Системна частина автоматизує такі процеси як: зберігання даних та товару, обробку даних та інформації на підприємстві, облік надходжень, вистежування надходжень, внесення змін в базу, оновлення даних в базі та в додатках. Друге та третє рішення розроблене для менеджменту підприємства, та написано для спеціально для двох найпопулярніших на даний час операційних систем: Android та iOS. Воно включає в себе системну та клієнтську частину. Системна частина контролює усі перелічені вище процеси підприємства, тобто життєвий цикл товару. Клієнтська частина дає можливість увійти в систему як адміністратор або користувач (менеджер) системи, виконувати навігацію по системі, тобто: переглянути інформацію про продукт, змінити її або додати нову, фільтр по: імені, оновленню, кількості; сканування штрих коду товару, додавання нового продукту та його штрих коду, сканування нового штрих коду, видалення товару або штрих коду з системи, корегування профілю користувача системи.

В написанні мною роботі, я ставила перед собою перш за все завдання створити систему, яка б змогла забезпечити коректну роботу підприємства з їхніми мінімальними фінансовими або людськими затратами.

Актуальність теми. На даний момент електронні обчислювальні машини дуже популярні в якості автоматизації будь якої інформації. Введення інформаційних баз даних стало вирішальним і дало можливість залишати в пам'яті машини лише ту інформація, яка потрібна для вирішення задач на які розрахована ця система. Дуже часто ми можемо зустріти такий термін як: СУС тобто Система Управління Складом, але деякі розробники систем відносять їх не до WMS, а навіть до IMS (Inventory Management System) або до WCMS (Warehouse Complex Management System) і т. д. Так як з кожним днем додаються нові технології, я вирішила що

доречним буде спроектувати систему до якої в перспективі можна буде додавати будь які модулі. Тобто ERP-систему. При згадці про систему управління, ті, хто більш-менш орієнтується в цій темі розуміє, що це можна описати такою термінологією як: термінали, етикетки, штрих-коди та інші атрибути впровадження. Ті, хто не орієнтується, асоціює «управління складом» з «складським обліком», хоча це різні речі. Така не точність перекладу термінології призводить до не правильної реалізації системи для клієнта. Поясню. Складський облік це – оформлення товаросупровідної документації, ведення документації підприємства, ведення інформації про залишки на складі та інше. Управління складом це – автоматична ідентифікація підприємства. В моїй системі я намагалася об'єднати ці два поняття, та полегшити таким чином роботу на підприємстві.

Мета та завдання проекту. Я поставила перед собою завдання полегшити кількість рухів на підприємстві. Наприклад: в систему не потрібно вводити нові данні, достатньо просто зчитати штрих код. Також достатньо лише невеликої кількості інформації про товар, щоб знайти його в базі підприємства за допомогою пошуку. Також кожне введення нової інформації буде оновлювати базу у системі та у додатках у режимі реального часу. БД постійно знаходиться у режимі неперервного оновлення.

Предмет дослідження. Технології та методи в роботі з великою кількістю даних в системі управління.

Наукова новизна отриманих результатів. Я пропоную технології, за будуть забезпечувати комплексний підхід у контролі та вистежуванні даних на підприємстві.

Апробація результатів роботи. Перше рішення системи було впроваджено у реальне підприємство, та на даний момент працює. Друге та третє рішення у режимі впровадження, воно тестується менеджерами підприємства.

1 ПОСТАНОВКА ЗАДАЧІ

Давайте уявимо інтернет магазин з доволі широким спектром послуг та товарів. Припустимо, клієнт замовляє доставку на «Сьогодні» та обирає дуже протилежні групи товарів: набір різнокольорових маркерів, електронну миш, чайний сервіз та кілька книг. Для зібрання цього замовлення працівнику потрібно не менше години, тому що не всі замовлення знаходяться на сусідніх полицях. Це і є одна з головних проблем на підприємствах: відсутність групування товарів, відсутність фільтру, сканування та відсутність системи обліку в цілому. Кількість інформації в наш час доволі велика, тому контроль інформації та потоку даних - одна з найбільших проблем на великих та середніх підприємствах. Аде це торкається не тільки документативного потоку інформації, але й фінансового потоку, потоку персоналу і т.д. Також автоматизація процесу збору замовлення позитивно впливає на втому персоналу (оператора)[4]. Науково, вже багато часу намагаються вирахувати що саме найбільш впливає на варіацію часу складання одного замовлення робочим. Але, поки хтось намагається науково це вирахувати, клієнти та власники малого та середнього бізнесу вимагають прискорити процес передачі готового замовлення, щоб виконувати ще більше замовлень та зменшити витрати на персонал. Завдяки цьому, є необхідність та навіть потреба у використанні систем автоматичної обробки даних які допомагають обробляти інформацію швидше та розповсюджувати її всі пристрої.

Наша система працює за такою схемою: оператор вводить інформацію у систему, а вона в свою чергу переформовує її та розповсюджує по всім додаткам. Оператор та користувач системи повинен використовувати спеціальні компоненти що б вносити данні в систему (наприклад камеру, що б зчитати штрих код або QR код) для забезпечення цілісності надходження інформації та створення єдиної бази даних для усіх додатків.

Головною нашою задачею, є правильне проектування системи, що б у перспективі була можливість додання будь-яких модулів на будь-якій мові

програмування (кроссплатформенність). Це забезпечить створення у майбутньому нових функцій та підтримання системи.

Розглянемо другу частину нашої системи – додатки. Система може автоматично генерувати данні для надходження до додатків, що допомагає швидко відстежувати рух товару на підприємстві та бачити усі зміни на підприємстві в режимі реального часу.

1.1 Автоматизовані системи управління

Давайте розглянемо основні для підприємства та для інформаційних систем в цілому, це допоможе нам краще розуміти що таке корпоративні інформаційні системи або КІС та автоматизовані інформаційні системи.

Об'єднання кількох підприємств і обумовлює собою термін «корпорація». Одними з головних вимог до компаній, які прагнуть вийти на назву «корпорація» є: централізоване управління інформацією по всій компанії для забезпечення єдиної бази даних по всіх складах, магазинах тощо. Корпорація має складну ієрархічну систему управління так як вона представляю собою багатопрофільну структуру.

Система повинна працювати як взаємопов'язаний комплекс певних елементів структури так як це і є в загальному вигляді поняття системи яка працює в інтересах компанії як єдине ціле. ІС – це система, яка за допомогою певних інструменті збирає данні , передає та обробляє їх, та надає цю оброблену інформацію працівникам будь-якого рівня з метою реалізації функцій управління та підтримання циклу на підприємстві. Структуру можна побачити на Рисунку 1.1: це сукупність окремих частин системи які працюючи разом називаються підсистемами.



Рисунок 1.1 – Структура Інформаційної системи підприємства

Корпоративна інформаційна система (КІС) – це інтегрована система управління підприємствам або корпорацією, яка зумовлює використання усіх даних підприємства та їх поглиблений аналіз. Також вона працює з документа-обігом та електронних діловодством. Тобто, вона веде повний цикл перебування товару на підприємстві, з його надходження, перебування, пересування та передачею. Вона реалізує та автоматизує ідеї та методи за допомогою інструментів та технічних засобів автоматизації. В неї також можна включити: технічне забезпечення підприємства, математичне, програмне, інформаційне, організаційне та правове забезпечення як зображено на Рисунку доданому зверху.

Основним завданням такої корпоративної системи, є коректне управління усіма ресурсами компанії що допомагає підвищити прибуток для власника корпорації та максимально задовольнити матеріальну сторону, тобто зменшити кількість зайвих рухів працівників та збільшити кількість продаж на підприємстві.

Систему управління можна тільки тоді назвати корпоративною інформаційною системою, коли вона включає в себе такі вимоги:

- Надійність та захист даних;

- Реалізовано віддалений доступ до системи та додатків;
- Є можливість подальшого супроводу системи, наявність відповідних інструментів;
- Інструментальні засоби для адаптації системи на іншу частину корпорації, наприклад при переносі даних на новий філіал;
- Інтеграція та консолідація нової інформації в систему;
- Можливість обміну даними з іншими програмними продуктами, системами, додатками і тд. тобто, можливість до написання модулів;
- Можливість аналізування стану системи, аналізування процесу експлуатації.

Корпоративна система також має доволі багато переваг впровадження:

- Власник підприємства або керуючий може оперативно отримати данні про стан підприємства, відслідкували переміщення товару. Завдяки додатку, він може це робото з власного смартфона;
- Перегляд поточних процесів в підприємстві;
- Висока ефективність управління підприємством завдяки постійному контролю та неперервному оновленню даних в системи;
- Зменшення часу збору замовлення та скорочення робочої операції;

Існують багато видів Інформаційних систем управління. Я обрала деякі з них, які підходять для впровадження в підприємство (деревообробна компанія) на якому на даних час працює моя система:

- CALS (Computer-Aided Logistic Support) – автоматизована підтримка для поставлення товару;
- SCM (Supply Chain Management) – контроль та керування циклом поставки;
- MRP (Material Requirements Planning) – оптимізація складського приміщення, фільтрування, розміщення по підходящим приміщенням, створення календаря «потреб» для замовлення у постачальника, оптимізація «запасів» підприємства;

- CAD (Computer-Aided Design) – допомагає автоматизувати проектування готового виробу на підприємстві;
- CAM (Computer-Aided Manufacturing) – управління інструментами проектування та розробку виробу, в тому числі і вестатами, станками, машинами для обробки деталей, металорізних верстатів, деревообробних машин тощо;
- CAE (Computer-Aided Engineering) – автоматизує усі види розрахунків, в тому числі і інженерні;
- PLM (Product Lifecycle Management) – керує життєвим циклом товару на підприємстві;
- MRP 2 (Manufacturing Resources Planning) – планування виробництва, контроль завантаження та відвантаження виробничих потужностей;
- CSRP (Customer-Synchronized Resources Planning) – працює зі споживачами, розплановує та контролює ресурси ;
- CRM (Customer Relationship Management) – менеджерую роботу з клієнтами.

Я обрала для себе за приклад створення ERP-системи. Enterprise Resource Planning System – це система, яка дає можливість з часом додавати в неї будь-які модулі. Вона об'єднує в собі усі ресурси підприємства та має принцип єдиної бази даних [3]. Це дає можливість контролювати усі бізнес-процеси. Також вона дає можливість одночасно робити декілька підключень, тобто одночасно працювати з кількома клієнтами (як товстими так і тонкими) в яких можуть будувати різні права та повноваження.

1.2 Сутність ERP систем

Системи класу MRP та MRP II в інтеграції з модулями FRP отримали нові функції та назву ERP-системи. ERP що від англ. Enterprise Resource Planning стали першими серед усіх систем на ринку, так як вони найбільше ефективно могли планувати діяльність підприємства. На Рисунку 1.2 нижче описані функції які

давали ті або інші системи. Згодом, з'явилися нові модулі та інтеграції, що надавали набагато більше можливостей та контролювали безпеку даних.

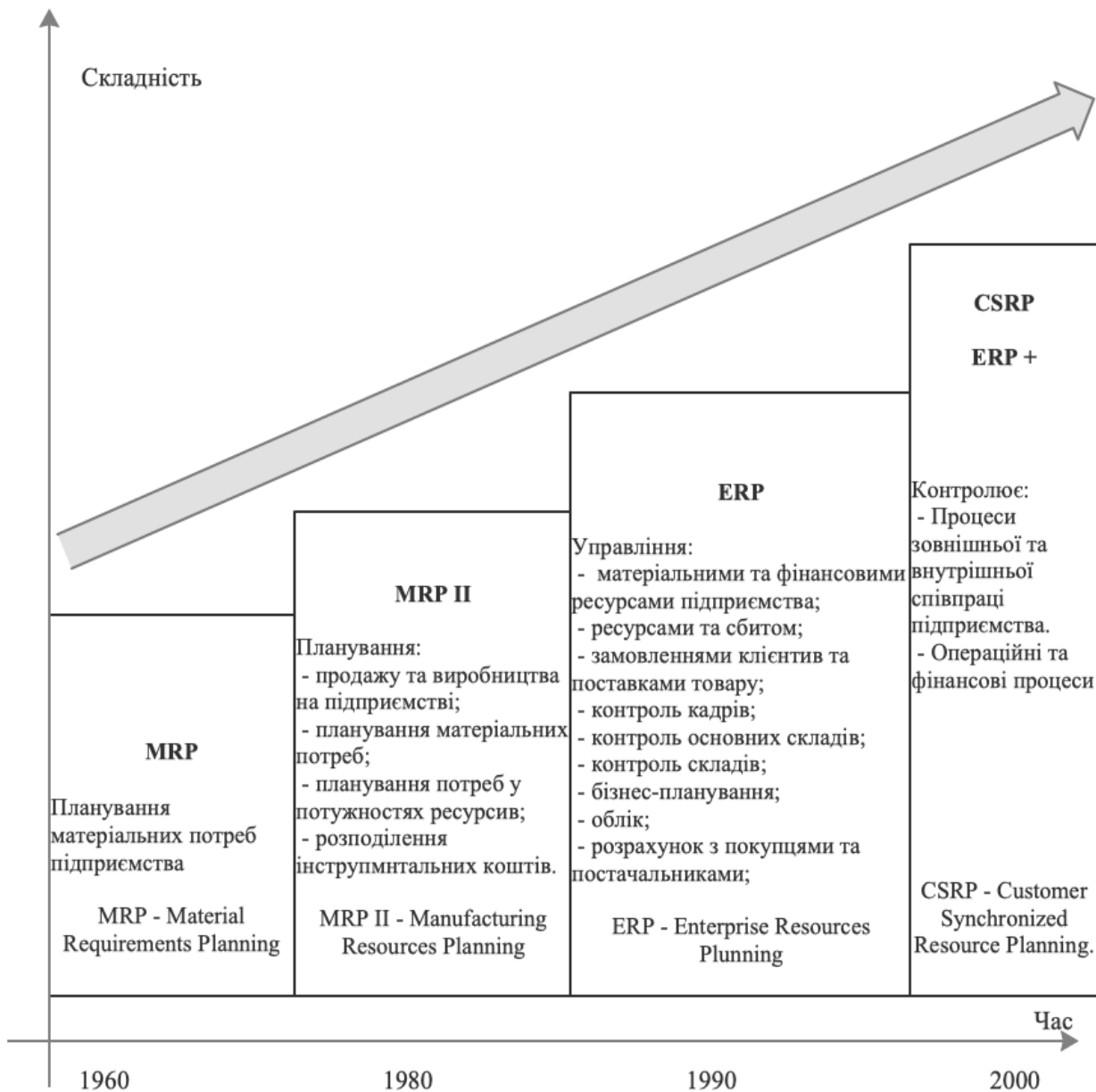
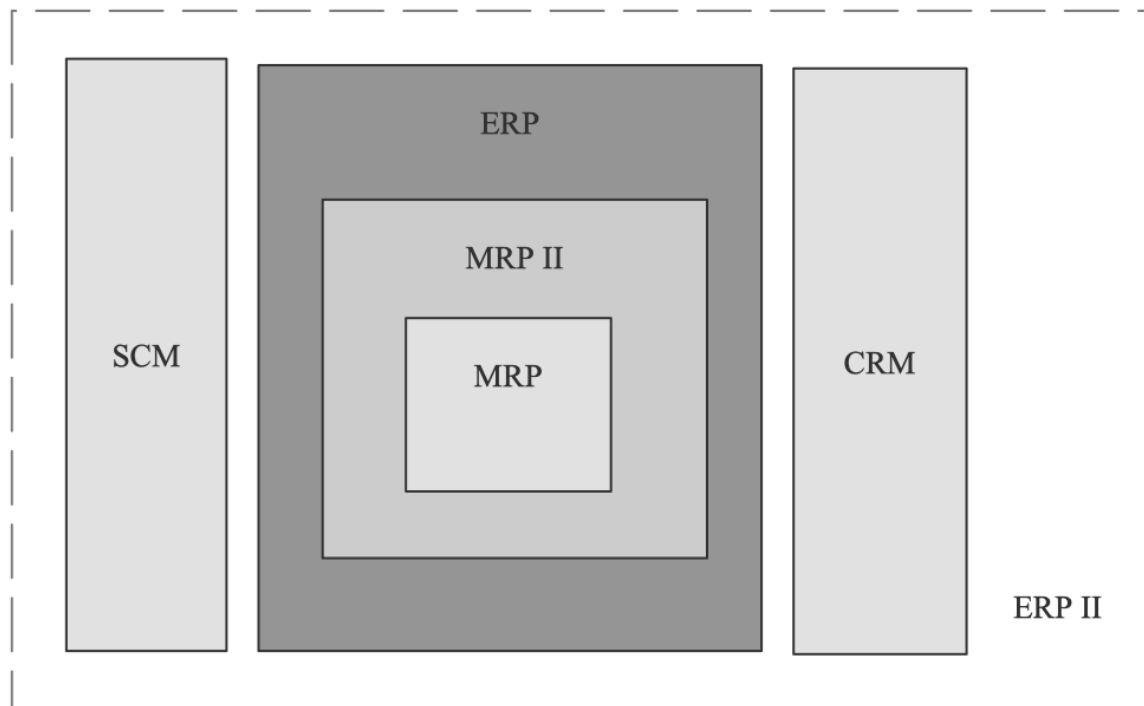


Рисунок 1.2 – історія розвитку ERP системи

Починаючи з 1999 року, розробники почали розвивати системи контролю та автоматизації підприємств, що дало можливість додати нові функції та модулі, оптимізувати бізнес процес підприємства та пришвидшити процес роботи[9]. Ідея створення ERP були в тому, що система могла працювати з усіма внутрішніми ресурсами фірми. Надалі, завдяки гнучкості, їй почали надавати нові права та

функції завдяки доданню нових модулів, таких як: SCM (дає можливість контролювати та управляти ланцюгами поставок), CRM (полегшує управління роботи з замовниками та клієнтами). Тож, з'явився новий тип системи ERP II (Enterprise resource and Relationship Processing) який міг відповідати за оптимізацію зовнішніх зв'язків підприємства а також контролювати внутрішні процеси.

Рисунок 1.3 – Система класу ERP II



Основні функції ERP II:

- Більш глибока функціональність;
- Завдяки новим інструментам, спростився процес створення вузькоспеціалізованих галузевих рішень;
- Між корпоративні бізнес-процеси удосконалилися та додалися нові моделі управління.

На основі викладеного мною вище, я виділила основні функції даної системи:

- Формування плану продажів та планів виробництва;
- Моніторинг потреб для підприємства;

- Управління та контроль закупівлями та залишками товару на підприємстві;
- Складання плану виробництва;
- Урегулювання та оперативна організація фінансової частини;
- Контроль, моніторинг та управління проектами.

ERP система проектується на основі трьох-рівневої клієнт-серверної архітектури. Приклад такої архітектури я зобразила на Рисунку 1.4. Сюди входять:

1. Рівень представлення: додання та виведення даних для оператора системи;
2. Рівень додатків: обробка та конвертація даних до додатків;
3. Неперервне оновлення даних, єдина база для усіх додатків системи, постійне резервування даних для уникнення їх загублення, розміщення даних на кількох серверах, найбільш захищений елемент.

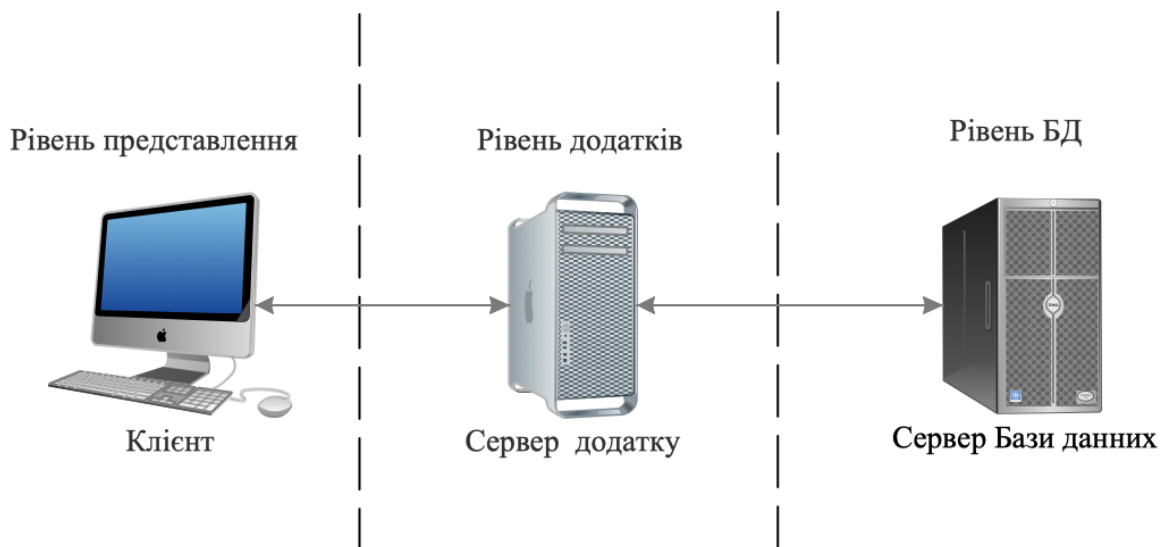


Рисунок 1.4 – Архітектура ERP системи

ERP має мереживну інфраструктуру, тому взаємодія проходить таким чином: в БД надходять данні про компанію (це може включати в себе документи бухгалтерського обліку, контракти, відомості по ним, кадровий облік і т.д.) та інші дані, які підходять для аналізу та враховують специфіку підприємства. Завдяки цьому, ERP система може коректно працювати, та швидко надавати будь яку

аналітичну інформацію. Для підприємств на українському ринку, дуже вадливим є питання платежів. Найчастіше система буде включати в себе ліцензію або покупку річної ліцензії тощо. На рисунку 1.5 зображено послідовність виконання в системі.

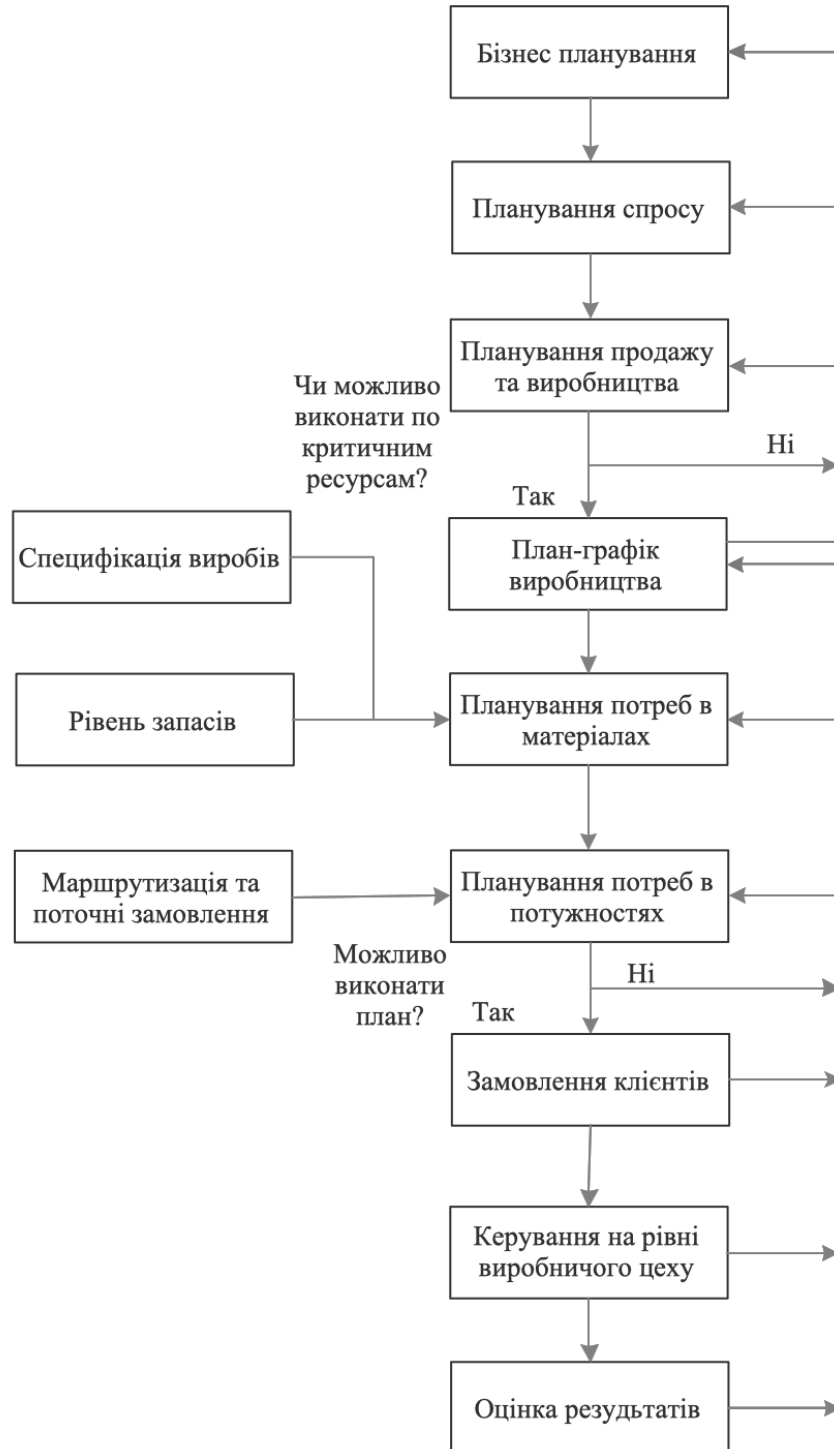


Рисунок 1.5 – Структура ERP

Тут наглядно можна побачити адекватну модель поведінки системи, що дозволяю створити якісну та правильну модель бюджетування, оцінити стан на

підприємстві та виробити план подальшої роботи. Усі розроблені ERP системи об'єднує однакова модель проектування.

- Платформа. Надає стандартні можливості для роботи модулів та компонентів. Розробник системи має доступ до програмного коду та може вносити зміни. До складу платформи входять: ядро (середовище до якого в перспективі можна буде додавати нові компоненти) та базовий функціонал (не може бути відключений, входить до систему за замовчуванням. В ньому знаходяться усі інструкції для користувачів системи, довідка т.д.);
- Керування даними. Обробка та інтерпретація даних під систему та для передачі їх у інші програмні модулі або додатки, база даних підприємства, положення даних на сервері, програмне забезпечення для роботи с базами даних;
- Модулі. Це будь-які компоненти які можна впровадити у систему та розширити її функціонал. Через те, що вони працюють незалежно один від одного, та з єдиною базою, не має проблеми оновлення всієї системи при підключенні нового модуля. Це одна з головних плюсів ERP системи. Їх можна поділити на такі типи:
 1. Модулі для роботи з зовнішніми клієнтами, зовнішніми користувачами, реальними та потенційними партнерами, постачальниками, покупцями тощо. Це наприклад онлайн магазин, особисті кабінети клієнтів, постачальників та менеджерів по роботі з клієнтами, кабінети ресейлерів. Також це може бути модуль, який перенаправляє на CMS систему для створення сайту або на конструктор.
 2. Конвектори беруть API з ядра, та зв'язує з зовнішнім додатком. Завдяки їм, можна додати телефонію, налаштувати обмін даних з додатком, сайтом і т.д.Дану структуру ми можемо побачити на Рисунку 1.6.

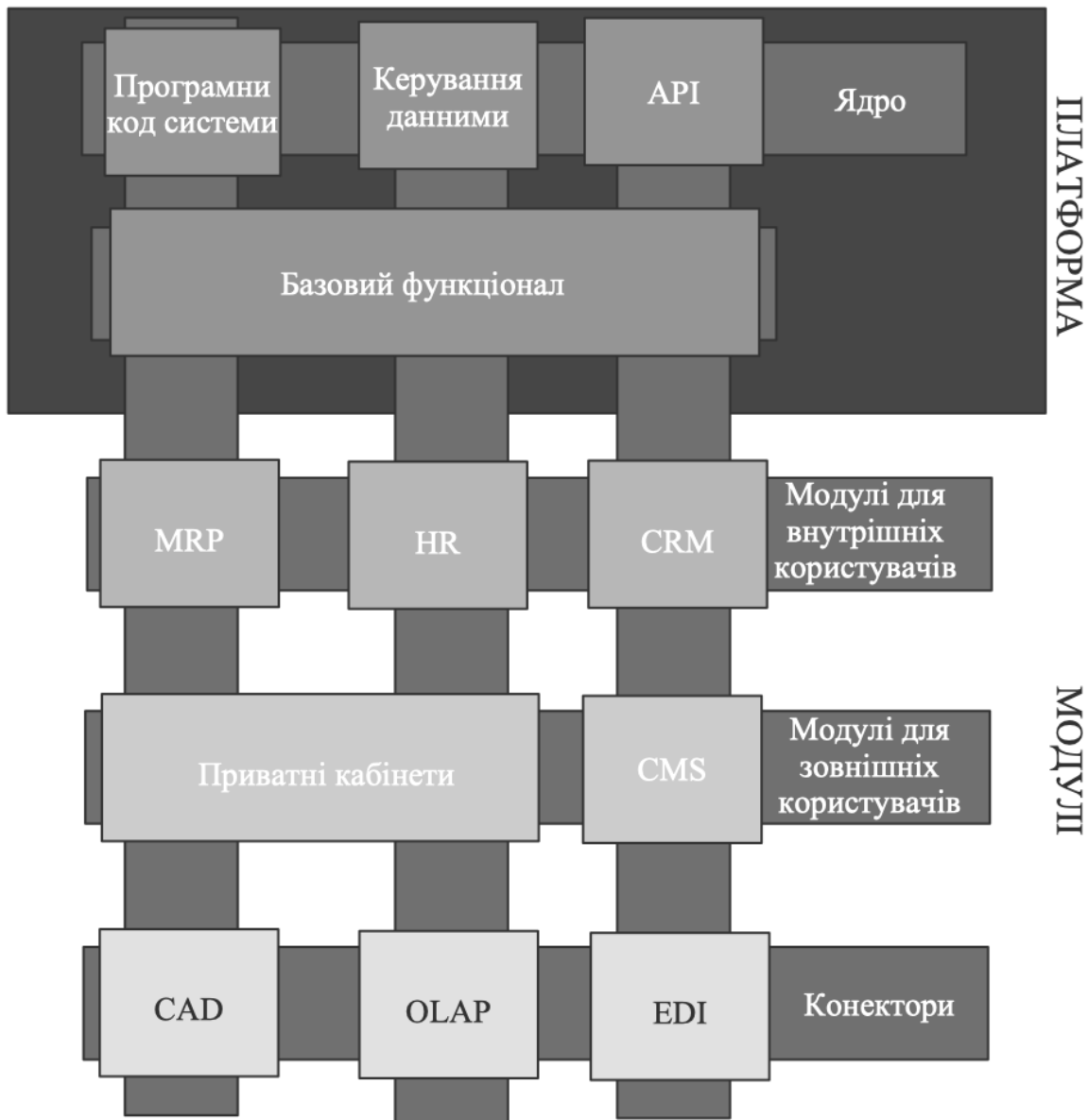


Рисунок 1.6 – Архітектура Платформи та Модулів ERP системи

Як можна побачити, конектори використовують EDI OLAP та CAD хоч вони і не входять до ERP[5]. Конектори можуть працювати з ними, тому що вони займаються тільки передачею даних та обміном інформацією. Архітектура моєї системи також модульна і передбачає в перспективі підключення нових інструментів.

Завдяки такій структурі, користувач отримує систему, яка буде працювати багато років адже в неї є дуже широкий перелік можливостей для розвитку. Власнику підприємства не потрібно змінювати програмний продукт, достатньо підібрати та підключити готові модульні рішення.

1.3 Аналіз підприємства

Давайте розглянемо підприємство типу «Наш зелений світ», та розпишемо які переваги несе собою впровадження WMF. ERP або іншої системи такого типу.

Для деревообробних компаній такого типу, дуже важливим є постійна підтримка клієнтської частини та швидкість виконання замовлення. Адже вони виробляють як і стандартні замовлення так і екстра замовлення: складі тендери, спеціальні конструкції, спеціальні ексклюзивні замовлення від клієнтів, оптові замовлення і т.д. під яке потрібно спеціально замовляти сировину, металеві частини, тобто працювати з іншими постачальниками. Отже, тільки щоденна пряма лінія зв'язку з клієнтами може оптимізувати швидкість роботи (адже в будь-яку хвилину клієнт може внести зміни в замовлення що невідкладно та оперативно треба ввести в систему), та покращити сервіс. Філософією компанії, є створення гнучких основ для роботи з зовнішніми клієнтами на ринку України, з партнерами з метою успішного виживання ринку та витримки конкуренції. Також компанія має оптові склади та магазини для внутрішніх продажів, комісійні магазини, складські приміщення які також повинні бути внесені в одну, цілісну базу даних.

Деревина, це не єдиний матеріал, який компанія надає в якості послуги, також це: гравірування по металу, по дереву, змішування фарб для надання спеціального кольору, введення в експлуатацію, навчання як користуватися спец обладнанням та технологіями, монтаж систем замикання гаражних воріт, замовлення запасних частин т.д. Також логістика або доставка здійснюється якомога швидше після отримання замовлення. Замовлення передається у відділ, там воно збирається, складається до вантажного автомобіля та відвозиться.

Все під одним дахом: 4000 квадратних метрів зберігання продукції, повний асортимент більше 10000 виробів, дверей, компонентів вставки, ключі, фарби, технологій замикання тощо. Тому База даних, а особливо безпека даних, для даного

підприємства є дуже важливим атрибутом! Також є постійна поставка матеріалів з Іспанії, Франції та Італії, що також треба враховувати в системи Автоматизації.

Для того, щоб визначитися з вимогами до нашої системи, давайте розглянемо намальовану мною структуру для підприємства типу «Наш зелений світ» на Рисунку 2.1.



Рисунок 1.7 – Структура підприємства типу «Наш зелений світ»

Керівництво підприємства такого типу може складатися з кількох або одного засновника:

- Виробничий відділ, робітники займаються проектуванням макетів дверей, цей відділ може складатися з архітектора та кількох інженерів-проектувальників;
- Технічний відділ, логістики, доставки та постачання, він відповідає за матеріально-технічне забезпечення, частіше за все може складатися з начальника відділу та кількох менеджерів;
- Відділ управління персоналом займається підбором персоналу, кадровим діловодством, найчастіше складається з ейчарів та кількох менеджерів;
- Відділ управління фінансами відповідає фінансову діяльність компанії, готує фінансову документацію.

1.4 Визначення необхідних функцій системи

Стає зрозумілим, що питання про актуальність програмного забезпечення ERP є зайвим. Звичайно, багато компаній покладаються на звичайні програмні програми, такі як Excel або Word, для написання правильних та візуально привабливих замовлень. Тим не менше, ці програми закривають розрив у вхідних платежах, відкриті елементи і працюють лише за допомогою великої кількості програмних та людських зусиль.

Сучасні, вигідні рішення корпоративного управління ERP можуть робити набагато більше[6]. Вони створюють прозорість із запиту клієнта, управління замовником і постачальником, покупки товарів, планування виробництва та персоналу, виробництва: з доставки, до документа про оплату. Натисканням однієї кнопки, прибутковість та будь-які недоліки у виробництві або розрахунку можна покласти на папір за допомогою модулю які переводить усі данні у txt-файл. Компанії швидко і легко відстежують вкладений капітал та отримують уявлення про рентабельність інвестицій. Повністю інтегрована система ERP також може спростити існуючі процеси та заощадити ващі ресурси. Наша система повинна працювати як під інтернет-магазин, замовлення якого надходять електронною поштою і не пов'язані безпосередньо з системою управління товарами.

Завдяки моему програмному рішенню, клієнти можуть покладатися на широкий спектр варіантів та галузевих рішень, адже завжди можна додати щось

нове до системи або усі елементи можуть з легкістю бути адаптовані для індивідуальних потреб завдяки інтегруючим модулям.

1.5 Принцип єдиної бази

Давайте уявимо компанію де облік ведеться за допомогою Excel. Складський облік використовує власну облікову систему, бухгалтерський облік – власну. Передача даних між цими підрозділами робиться за допомогою паперових файлів або в усній формі – що значно зменшує швидкість роботи підприємства або дані можуть бути передані не точно. Після чого потрібно знову внести данні до потрібної системи обліку. Всі ці фактори дуже сильно затримують передачу даних, це не припустимо в наш час коли обсяг інформації збільшується з кожною хвилиною. Через це виникає необхідність використовувати автоматичні хмарові або серверні бази даних.

Принцип єдиної бази, це:

1. Контроль;
2. Віддалене керування;
3. Точність передачі даних;
4. Оперативність.

Авторизація в базі робиться на рівні сервера, через ролі логінів і серверів, а також через користувачів бази даних та ролей в ній. Модель обраної мною бд SQL, забезпечує однакову систему даних в кожній базі даних. Отже, ми маємо дані в базі даних, а з SQL ми можемо вибирати дані, вводити нові дані, оновлювати чи архівувати уже існуючі дані з бази.

MySQL - це лише одна з багатьох баз даних, але яка широко використовується в малому та середньому бізнесі. База частково безкоштовна і з легкістю працює з PHP. PHP та MySQL доступні в більшості пакетів веб-хостингу. SQL - це мова запиту бази даних.

Основним принципом безпеки є – назва ідентифікацій, які використовують MySQL Server. Це, як правило, це люди або групи людей, але вони також можуть бути іншими суб'єктами. Керування безпекою може бути реалізовано за допомогою

переліченого Transact-SQL або за допомогою SQL Server Management Studio[7]. Логін допомагає контролювати вхід до бази за допомогою облікових записів користувачів. SQL Server і SQL підтримують вхідні дані на основі автентифікації Windows і SQL Server.

Крім того, швидкість передачі даних з такою системою – миттєва. Оператор зразу бачить нове замовлення та передає завдання іншому підрозділу. Це допомагає скоротити час, та збільшити кількість виготовлених замовлень в день.

1.6 Урахування змін ринку

Оскільки сьогоденні компанії дуже складні, вище керівництво все більш часто вирішує виконувати всі завдання планування і контролю автоматизованої ERP-системою. Особливо в плані конкурентоспроможності швидкість відіграє все більш важливу роль в наданні інформації. Це можливо тільки з інтелектуальним програмним забезпеченням і потужним обладнанням та можливістю додавати нові віджети та інструменти. Через їх здатності своєчасно складати карту компанії в цілому, системи планування ресурсів підприємства стали незамінними в сучасній практиці прийняття рішень.

Як проекти системи можуть зазнати невдачі? Це через недостатню підготовку або складності самої системи ERP? Не тільки. У великих компаніях існує багато обов'язків, залежностей які несуть загрозу в результаті впровадження системи. Співробітники бачать, що їх важливість зменшується, адже все більше процесів автоматизується. Дотримуючись існуючих структур та інструментів, співробітники також можуть критично оцінити впровадження системи. Багато хто боїться труднощів і вважають за краще залишатися з та працювати з стареньким Excel.

Для того щоб планування загально організаційних ресурсів могло ефективно виконувати свою роль інструменту навігації і управління в більш складних середовищах, використовуються галузеві програмні рішення ERP або WMS

2 ОГЛЯ ІСНУЮЧИХ РІШЕНЬ

Тенденцію рішень в галузі обробної промисловості можна прослідити вже зараз. Хоч вони раніше були внутрішніми, сьогоднішні системи взаємо з'єднують з всією інформацією по всьому ланцюжку постачання. Не тільки прямі постачальники, але й внутрішні постачальники компанії тепер інтегровані. Таким чином зміни умов виробництва або доставки можуть бути заздалегідь передбачені та враховані при плануванні робочого процесу. Це дозволяє зробити усі процеси на підприємстві прозорими.

SAP



Рисунок 2.1 – Десктопний та мобільний інтерфейс SAP

SAP була розроблена на ABAP, мовою програмування SAP. Вона є одним з провідних світових постачальників корпоративних програмних рішень як на території Німеччини, так і на території Європи. Вона організовує різні процеси всередині підприємства і між компаніями. Система включає в себе бізнес-додатки для великого і середнього бізнесу, а також стандартні рішення для малих і середніх компаній[13]. Крім того, система SAP є галузевими рішеннями та підтримує

основні процеси в сфері торгівлі, фінансів, високих технологій та державного управління. Флагманом групи є SAP Business Suite, який може бути точно адаптований до відповідних вимог і бізнес-цілей. Основою цієї програми є технологія бази даних Hana, розроблена SAP, в якій дані більше не зберігаються на жорсткому диску.

SAP має широкий спектр вбудованих стандартних функцій. Наприклад, в процесах «Продажі», «Відвантаження» або «Управління запасами» дані автоматично переносяться в функції обліку. Додатки SAP Business Suite допомагають індивідуально керувати найважливішими бізнес-процесами. В цілому, вони надають тісно інтегроване пакетне рішення.

Хоч вищезгадані рішення для багатьох малих підприємств значно знизили б поріг для використання системи ERP, справжнім проривом є ERP хмарні, які стали доступними лише протягом декількох років. Це програмне забезпечення як сервісне рішення, де користувач може отримати доступ до готового програмного пакету ERP онлайн і використовувати його для планування і контролю ресурсів. Незалежно від того, в яких галузях активний користувач - в разі хмарної ERP витрати набагато більш точно розраховуються і контролюються, ніж витрати на локальні рішення.

Крім того, постачальники надають багаторівневі рішення для захисту даних, які у багато разів перевищують рівень безпеки на рівні підприємства. Для малих підприємств, які не можуть дозволити собі ні програмного забезпечення, ні обладнання, не кажучи вже про IT-відділ, це може бути кращим інструментом.

1. обчислюються витрати
2. висока безпека даних
3. не потрібен власний IT-відділ

Weclapp

Weclapp – німецька компанія з Марбурга. Всі дані зберігаються в Німеччині відповідно до строгих німецьких правил захисту даних.

З головних плюсів, є швидке впровадження, що зробило цю систему – вибором 2018 року. Weclapp підтримує всі сфери бізнесу: CRM, ERP-систему,

програму обліку та білінгу. Використання єдиного рішення забезпечує значну економію часу та коштів, а також прозорість а роботі. Також, система має інтуїтивно зрозумілий і простий у використанні сучасний, призначений для користувача, інтерфейс. Вбудована вкладка з допомогою для користувача і рекомендаціями щодо використання, роблять роботу з програмним забезпеченням дуже простим.

Оскільки weclarr задуманий як «хмарна система», витрати на впровадження відсутні. Також виключаються витрати на обслуговування, так як усі оновлення безкоштовні. Завдяки модульній конструкції, ви платите тільки за те, що насправді потрібно вашому підприємству.

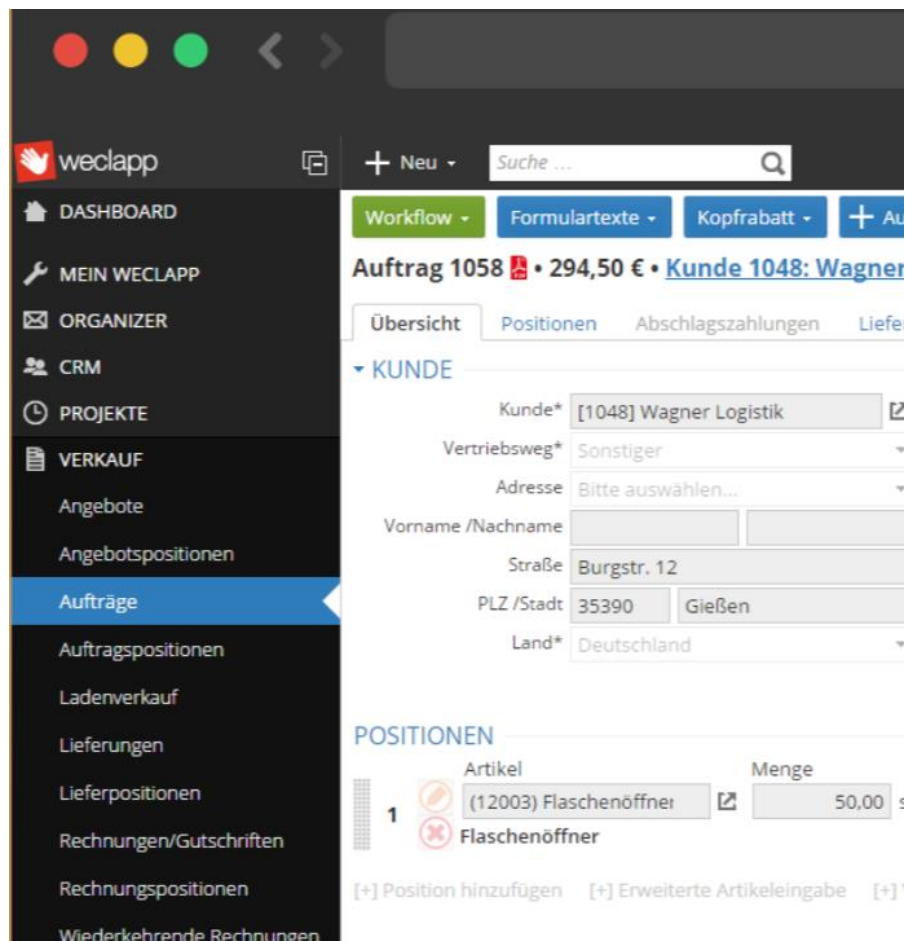


Рисунок 3.2 - Рисунок 2.2 – Інтерфейс німецької системи WeClarr Actindo

Переосмислення програмного забезпечення ERP: Actindo об'єднує всі процеси торгівлі, доставки, складування і логістики в одну централізовану рішення API First Platform.

Значно покращений час завантаження, навігації та панелей моніторингу якщо порівнювати с іншими системами. Нова платформа Actindo

Core 1 з функцією масштабування забезпечує оптимальну продуктивність для майбутнього зростання. Рішення Actindo IQS 2.0 забезпечує ефективне управління окремими процесами доставки і маршрутизації, включаючи нові засоби управління MDE на базі Android. Значно простіше інтегрувати додаткові канали поширення (тобто додатки для платформ Android або iOS) через підхід API-first. Також, завдяки готовими до установки рішеннями в Apple Store, ви можете розширити Actindo бізнес-додатками і компонентами для кожного відділу і галузі.



Інтерфейс системи зображено на Рисунку 3.3.

Рисунок 2.3 – система автоматизації Actindo

3 РОЗРОБКА СИСТЕМИ АВТОМАТИЗАЦІЇ

Нижче я перерахую вимоги до систем, що б додаток працював правильно та коректно. Рекомендовано встановлювати останні версії прошивок та оновлень.

3.1 Формування вимог до системи

Процесор:

iOS – не має значення;

Android – KitKat 4.4 або пізніша версія, також ARM або Intel x86;

Пам'ять:

iOS – не важливо;

Android – 1 ГБ;

ОС Windows – повинні підтримувати підтримувану версію Windows 10 Mobile.

Планшети:

iOS – вимагають iOS 11.0 або новішої версії. Для iPad Pro потребує iOS 11.0 або новішої версії. Додаток підтримується за останніх двох версій iOS. Коли вийшла нова версія iOS, вона стане поточною версією. Моя служба підтримується відповідно до цієї та попередньої версії iOS.

Android – систему можна встановити на планшетах і смартфонах, які працюють на підтримуваний версії Android, і мають процесор Intel ARM або процесор Intel x86.

Підтримка пристроїв Kitkat та Lollipop триватиме тільки квітня 2019 року.

Функціональність та доступність функцій продукту можуть відрізнятися залежно від старих систем[10]. Щоб забезпечити найкращий спосіб роботи з користувачами, скористайтеся найновішою версією зазначених вище операційних систем. Функціональність продукту та продуктивність графіки можуть також відрізнятися залежно від конфігурації системи. Для деяких функцій може

знадобитися розширення або підключення до сервера. Для ліцензування та доступу до послуг потрібен обліковий запис.

Щоб повністю скористатись можливостями сканування БАР коду або штрих коду, вам потрібна стандартна вбудована камера або USB 2.0 відеокамера.

Персональний комп'ютер на якому буде знаходитися система повинен включати в себе: миш, клавіатуру, принтер, MS Word, порт для usb та для інших передавачі даних.

Вимоги:

- 2 Гб пам'яті на жорсткому диску;
- Windows 2010 або не пізніше 8;
- Процесор з частотою 600МГц;
- 256 МВ оперативної пам'яті;
- MS Word 2010 або вижче.

3.2 Логіко – структурна матриця

Для реалізації системи, яка автоматизує складський облік на підприємстві, вирішено було спроектувати логіко-структурну матрицю[14]. Для її впровадження я виділила 6 зацікавлених сторін:

- 1) Керівництво підприємства. Генеральний директор компанії, який фінансує підприємство. Вигодою для даної ролі буде успішна реалізація проекту, збільшення прибутку в перспективі.
- 2) Користувачі системи. Співробітники компанії, які працюють з системою кожний день. Вигодою буде дружні інтерфейс, потрібних дій для виконання ряду операцій.
- 3) IT-відділ компанії. Готує підприємство для впровадження системи, готує прилади для провадження нових інструментів, займається навчанням співробітників новими діями, підтримкою співробітників для роботи с з системою. Працює з оновленнями.
- 4) Керівник проекту. Займається взаємозв'язком з компаніями-клієнтами, керує своїми проектами. Вигода – прибуток, завдяки швидкій та успішній реалізації проекту.

5) Конкуренти підприємства. Інші компанії, які працюють на тому ж самому ринку. Зацікавлені в інформації про нові можливості автоматизації підприємства.

На цьому етапі автоматизації була створена логіко-структурна матриця (Таблиця 3.1) проекту. Матриця відображає цілі, які ставить перед собою команда проекту.

Рівні цілей	Вимірні показники досягнення цілей	Джерела і методи для підтвердження досягнень	Припущення та ризики
Загальні цілі			
Впровадження інформаційної системи в підприємство «Наш зелений світ».	Наявність готової інформаційної системи для впровадження.	Так/Ні.	Не коректне впровадження системи. Втрата даних.
Конкретні цілі			
Розробка проекту для впровадження системи. Розробка системи. Впровадження. внесення змін в кінцеву версію за проханням керівництва.	Розроблений проект по впровадженню системи. Робоча система.	Так/Ні.	Недостача в інформації про процеси в підприємстві.
Очікувані результати			

<ol style="list-style-type: none"> 1. Запуснення в експлуатацію систему. 2. Підвищення кваліфікації співробітників (користувачів системи). 	<ol style="list-style-type: none"> 1. Функціонує система. 2. Кількість навчених співробітників, які можуть працювати з нею. 	<ol style="list-style-type: none"> 1. Так/Ні. 2. Звіти про проведення тренінгів для співробітників та тестування користувачами. 	<ol style="list-style-type: none"> 1. Припинення фінансування. 2. Брак мотивації у співробітників підприємства. 3. Відмова керівництва у впровадженні.
Заходи	Ресурси	Ціна ресурсів	Риски та припущення
<ol style="list-style-type: none"> 1. Складання вимог до системи. 2. Розробка ТЗ. 3. Налаштування системи з відповідністю до вимог. 4. Тестування системи. 5. Навчання користувачів: розробка інструкцій користувача, проведення тренінгів, 	<ol style="list-style-type: none"> 1. Ліцензія на ПО. 2. Обладнання для коректної роботи системи. 3. Персонал: команда проекту, учасники з сторони підприємства. 	<p>У стадії обговорення.</p>	<ol style="list-style-type: none"> 1. Чітке представлення про поточні процеси в підприємстві. 2. Усі поточні процеси коректно описані та задокументовані. 3. Поява нових користувачів системи. 4. Зміна команди підприємства. 5. Недостача коштів.

тестування користувачів.			
-----------------------------	--	--	--

Таблиця 3.1 – Логіко структурна матриця проекту

4 РЕАЛІЗАЦІЯ БІЗНЕС ЛОГІКИ

4.1 Описання алгоритму реалізації

Реалізація даної системи на підприємстві буде полягати у створенні програмного забезпечення, за допомогою якого користувач зможе з легкістю проводити операції. Класична система такого типу, є статичною та націлена на збір інформації з бізнес процесів без даних та аналізу цих даних. Сучасні ERP або WMS системи, навпаки, засновані на вимогах користувачів.

Доведено, що системи такого типу, підвищують продуктивність і знижують витрати на персонал. На відміну від класичного управління товарами, вони відображають всі сфери і бізнес-процеси компанії. У центрі уваги не лише управління матеріальними ресурсами, але також фінанси і бухгалтерський облік, управління персоналом, продажу, маркетинг, дослідження та інші сфери діяльності компанії. Функціональні розмежування, які раніше були загальними для логістики, фінансового обліку та контролінгу, наприклад, усуваються за допомогою комплексної системи. Всі області спілкуються між собою і використовують одну і ту ж базу даних.

Управління матеріальними ресурсами або планування ресурсів підприємства - це планування, управління і контроль всіх рухів матеріалів в компанії, а також між компанією та іншими суб'єктами бізнесу, такими як клієнти і постачальники.

Поняття матеріалу є широким, і описує як закупка товарів для перепродажу, сировину і напівфабрикати, необхідні для виробництва.

Їх необхідно закуповувати, зберігати і відправляти таким чином, щоб вони були наявні в достатній кількості, з необхідною якістю, в потрібний час і в потрібному місці. Давайте розглянемо процес поставки нового товару на підприємство та процес переміщення товару по складу на Рисунках 4.1 та 4.2.

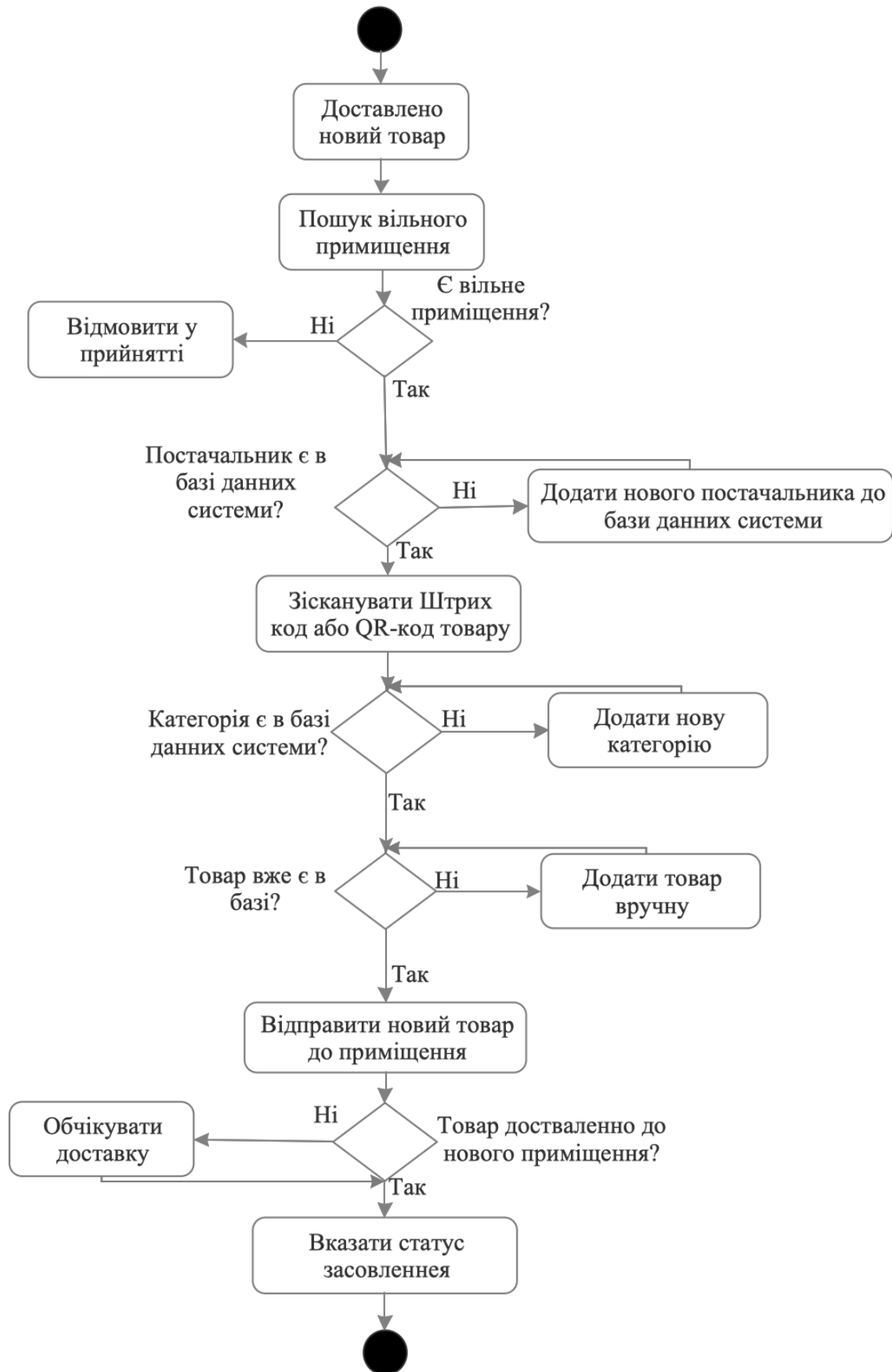


Рисунок 4.1 - Процес доставки нового товару на склад

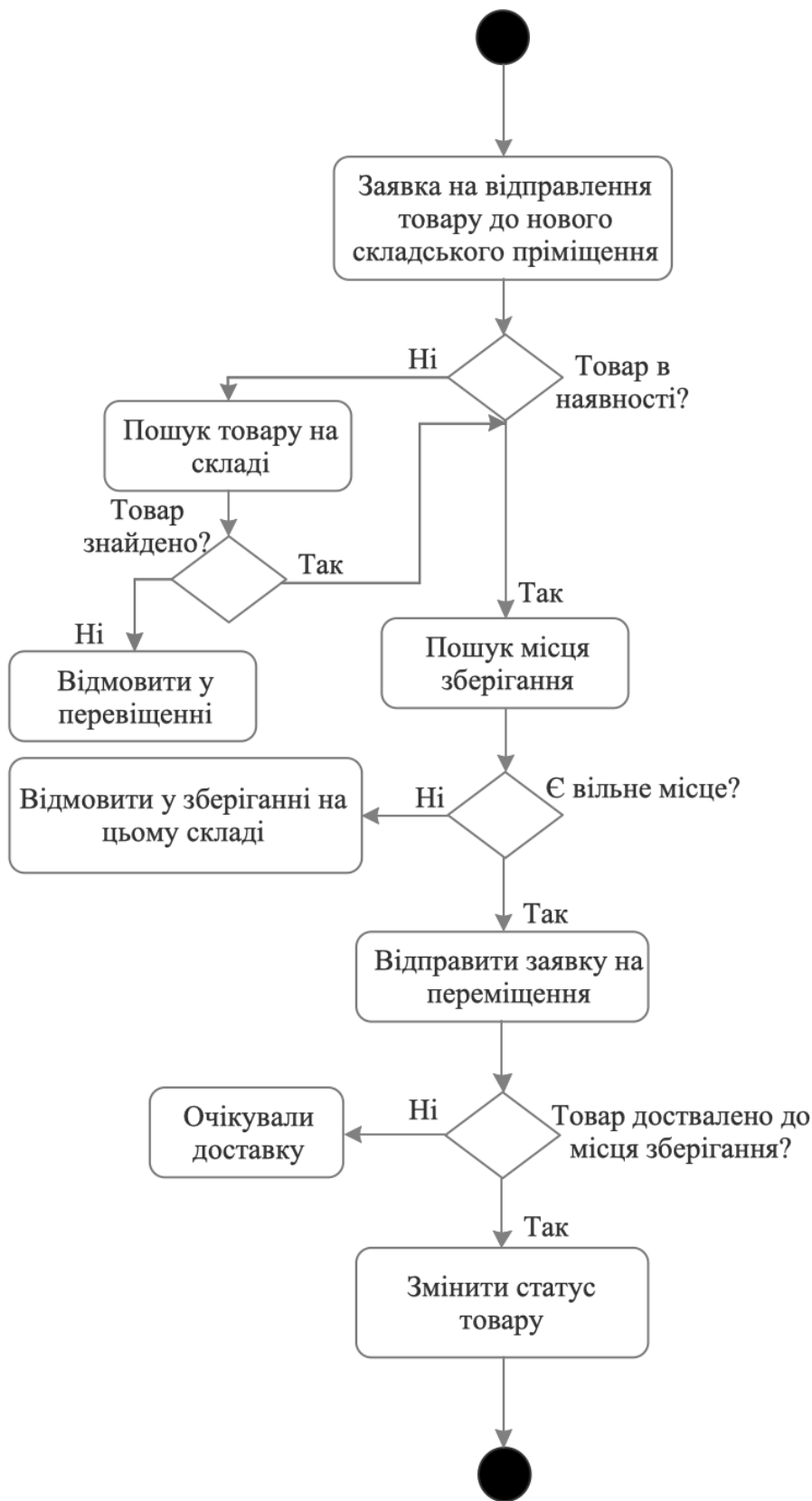


Рисунок 4.2 – Переміщення товару між складами

Коментуючи ці діаграми, я можу сказати що навіть рухи, які з першого погляду може здаватися не дуже важливими, будуть враховуватися системою. Тут можна також побачити як взаємодіють дві системи: десктопна версія системи, через яку, наприклад, оператор приймає новий товар, та версія для смартфона, завдяки якій можна зчитати Бар-код та додати новий товар до бази даних підприємства.

4.2 Модульна структура системи

В представленій нижче таблиці 4.1 – розписані основні функції системи.

Назва функції	Вхідні параметри	Опис
clearBase	-	Обнулити БД.
addItem	Name, State, User_id, Plasement_id, Type_id	Додати нову одиницю товару.
addTransaction	Name, Item_id, User_id	Додати нової транзакцію.
getAllTransactions	-	Отримати всі транзакції.
getTransactions	Id	Отримати конкретну транзакцію.
getItems	-	Отримати всі товари.
getItemsByClientId	Id	Отримати все товари відправлені до вибраного клієнта.
getItemById	Id	Отримати конкретний товар.

Таблиця 4.1 – Модульна структура системи

Назва функції	Вхідні параметри	Опис
addItemType	Name	Отримати тип конкретного товару.
getItemTypes	-	Отримати всі типи товарів.
addPlacement	Name, User_id	Додати нове приміщення.
getPlacements	-	Отримати всі приміщення.
updateClient	Id, Name, Contacts	Оновити/редагувати обраного клієнта.
addClient	Name, Contacts	Додати нового клієнта
getClients	-	Отримати список усіх клієнтів.
updateProvider	Id, Contacts Contract	Оновити/редагувати обраного поставника
addProvider	Name Contacts	Додати нового потсавника
getProvider	-	Отримати список усіх поставників
updateItem	Id, Name, State, User_id, Placement_id, Type_id, Client_id	Оновити/редагувати обраний товар.

Таблиця 4.2 – Модульна структура системи

Назва функції	Вхідні параметри	Опис
getProvider	-	Отримати список усіх поставників
updateItem	Id, Name, State, User_id, Placement_id, Type_id, Client_id	Оновити/редагувати обраний товар.
getItemName	Id	Отримати ідентифікаційний номер товару.
getUserName	Id	Отримати ідентифікаційний номер користувача.
getPlacementName	Id	Отримати ідентифікаційний номер приміщення.
getClientName	Id	Отримати ідентифікаційний номер клієнта.
getTypeName	Id	Отримати ідентифікаційний номер категорії.
getUsers	-	Отримати всіх користувачів.

Таблиця 4.3 – Модульна структура системи

5 РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

5.1 Реалізація клієнт серверної взаємодії

Модель клієнт-сервер використовується для розподілу завдань у мережі. Ця концепція надає користувачеві багато переваг, перерахованих нижче:

Центральне адміністрування: сервер знаходиться в центрі мережі, щоб він мав доступ до всіх користувацьких ресурсів, наприклад може керувати центральною базою даних. Технічне обслуговування також дуже просте: оновлення програмного забезпечення, наприклад, виконується тільки на серверах, так що клієнти зазвичай не помічають.

Економія ресурсів: оскільки дані зберігаються централізовано та доступні в будь-який час, не потрібно зберігати ці самі дані на різних клієнтських комп'ютерах.

Більша безпека доступу: централізоване зберігання даних дозволяє легко контролювати доступ. Перед доступом до певних даних клієнти повинні пройти аутентифікацію на сервері.

Розширювана мережа: додавання та видалення клієнтів можливе, не впливаючи на роботу мережі та не вимагаючи серйозних змін. Крім того, кількість клієнтів в принципі може бути продовжена без обмежень.

Не залежить від місця розташування. Через централізоване зберігання даних користувачі не пов'язані з певним місцезнаходженням, тому можливості програми дуже гнучкі.

Висока надійність: сервери дуже відмовостійкі завдяки системі RAID (надлишковий масив незалежних дисків).

Архітектура C / S - це системний дизайн, в якому обробка програми поділяється на дві окремі частини. Одна частина працює на сервері (бек-ендальний компонент), інша - на робочій станції (клієнт або інтерфейс). Обидві частини об'єднуються в мережі через мережі. Клієнт зобов'язується обробляти дані на сервері та використовує послуги сервера. На відміну від архітектури на базі хоста, сервери більше не зайняті всією обробкою даних, але повертають дані клієнту для подальшої обробки.

5.2 Desktopна частина системи

Controller – бібліотека, яка містить головні сутності додатку. Http протокол забезпечую взаємодію з усіма частинами ПЗ.

ПЗ складається з двох частин:

- клієнт – додаток для користувача;
- сервер – WCF сервіс;
- частини розробленого розподіленого ПЗ взаємодіють через http протокол.
- API сервер для додатку

В WcfServiceLibrary було реалізовано взаємодію з ContractServer та прописано у Program.cs. Було зроблено налаштування кінцевих точок, binding, а також інших параметрів клієнт-серверної взаємодії для коректної роботи додатку. Важливі поля, які часто використовуються у кожному з класів буде описано нижче.

Клас Client

Зберігає інформацію про клієнта. Перелік полів:

- Id (int) — унікальний ідентифікатор;
- Name (string) — ім'я або назву компанії клієнта/замовника;
- Contacts (string) — зберігає контактні данні клієнта.

Клас Item

Зберігає данні про товар. Перелік полів:

- Id (int) — ідентифікаційний номер товару;
- Name (string) — поле яке містить назву товару;
- State (string) — поле, яке містить статус даного товару (нове, б/у, відправлено клієнту);
- Type_id (int) — унікальний ідентифікаційний номер типу товару;
- Placement_id (int) — унікальний ідентифікаційний номер приміщення де зберігається даний товар;
- User_id (int) — унікальний ідентифікаційний номер відповідального за даний товар;

- Client_id (int) — унікальний ідентифікаційний номер клієнта якому було відправлено даний товар;
- Date (string) — дату здійснення останньої операції.

Клас Item_Type

Зберігає данні про тип товару. Перелік полів:

- Id (int) — унікальний ідентифікаційний номер товару;
- Name (string) — назву типу товару.

Клас Placement

Зберігає інформацію про приміщення в якому зберігаються товари. Перелік полів:

- Id (int) — унікальний ідентифікаційний номер приміщення;
- Name (string) — назву приміщення;
- User_id (string) — ідентифікаційний номер відповідального за приміщення.

Клас Transaction

Містить дані про транзакцію яка була здійснена з даним товаром. Перелік полів:

- Id (int) — містить унікальний ідентифікаційний номер транзакції;
- Name (string) — містить назву транзакції (наприклад: товар додано, товар переміщено, змінено стан товару, товар відправлено клієнту);
- Item_id (int) — містить ідентифікаційний номер товару з яким здійснено дії;
- User_id (int) — містить ідентифікаційний номер менеджера який здійснив операцію;
- Date (string) — містить дату здійснення операції.

Клас User

Містить в собі дані про користувача

- Id (int) — унікальний ідентифікаційний номер працівника;
- Name (string) — містить повне ім'я певного менеджера;
- Age (int) — містить в собі вік менеджера;
- Date (string) — містить дату прийняття працівника до штату складу.

Розробка додатку було реалізовано за допомогою платформи Visual Studio та на мові C# [1]. Даний тип системи передбачає дружній інтерфейс, інтуїтивно

зрозумілий інтерфейс.

Вхід в систему починається з вікна входу. Не зареєстрований користувач не може увійти в систем, але для нього є можливість перейти з цього вікна у поле для реєстрації. Поля в данному вікні (Логін та Пароль) є обов'язковими для заповнення. Якщо одне з полів заповнене некоректно, чи не заповнене взагалі, то реєстрація не відбудеться. Процедуру реєстрації нового користувача системи зображено на рисунку 5.1.

При наступному відвідуванні зареєстрований користувач має пройти процедуру авторизації, ввівши дані, що вказувалися під час реєстрації. Дані про користувача зберігаються у базі даних, та надаються до інших додатків.

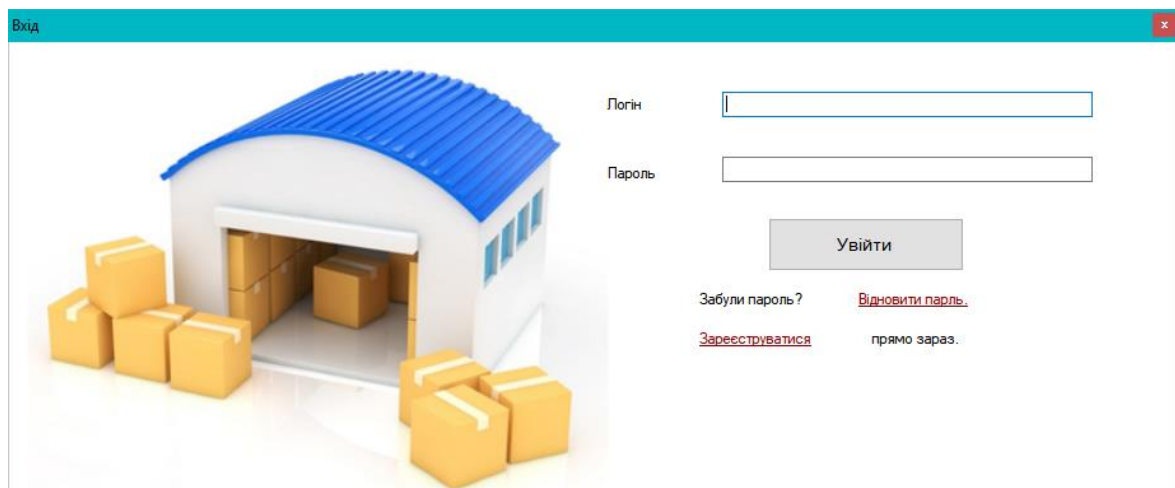


Рисунок 5.1 – Форма реєстрації

Після проходження процедури реєстрації та підтвердження нового зареєстрованого працівника адміністратором, він зможе зайти в систему вже через вікно входу.

Після проходження процедури входу, користувач може користуватися системою. Інтерфейс головного вікна містить в собі вікна та клавіши. Також вікно містить поле “Пошук” де користувач може зробити пошук за ID, назвою, тіпо, менеджером, датою та приміщенням. Пошук буде відбуватися по всі додаткам системи. Головне вікно системи зображено на рисунку 5.2.

Form1

WAREHOUSE

Введіть дані

Пошук

16 декабря 2018 г.

Адміністратор

Система управління складським обліком

ID товару	Назва товару	Стан	Тип товару	Приміщення	Оператор	Дата
1	Компьютерный стол MaximuM Maxi Komp 803 Matte	новый	Мебель	Главный склад	Иванов Вас...	16.12...
2	Стол Rondi Пилот-1 Ольга	новый	Мебель	Главный склад	Каргаполов ...	16.12...
3	Компьютерный стол + тумба Star - М Джуниор	Отпра...	Мебель	Главный склад	Каргаполов ...	16.12...
4	Обеденный стол Vetro Mebel TM-51 - 1 Белый	новый	Мебель	Главный склад	Иванов Вас...	16.12...
5	Стол Rondi Кухня-2 Дуб Сонома	новый	Мебель	Главный склад	Каргаполов ...	16.12...
6	Стул Примтекс Плюс 1011 alum S-64 Светло - бежевый	новый	Мебель	Главный склад	Каргаполов ...	16.12...
7	Стул Vetro Mebel M-11 Гаванна(M - 11 - gavanpa)	новый	Мебель	Главный склад	Каргаполов ...	16.12...
8	Стул Special4You Sedia White(E5746)	Отпра...	Мебель	Главный склад	Дягтерев Д...	16.12...
9	Стул Vetro Mebel M - 21 Синий(M - 21 - blu)	новый	Мебель	Главный склад	Иванов Вас...	16.12...
10	Табурет Новый Стиль Teddy Chrome V-19	новый	Мебель	Главный склад	Каргаполов ...	16.12...
11	Стиральная машина полногабаритная LG FH0B8QD	новая	Бытовая техника	Сортировочн...	Дягтерев Д...	16.12...
12	Газовая плита HANSA FCGW52028	Отпра...	Бытовая техника	Сортировочн...	Иванов Вас...	16.12...
13	Газовая плита GEFEST 3200 - 06 K19	новая	Бытовая техника	Сортировочн...	Дягтерев Д...	16.12...
14	Газовая плита GORENJE GN 5112 WF	новая	Бытовая техника	Сортировочн...	Дягтерев Д...	16.12...
15	Однокамерный холодильник VESTFROST VD142RS	новый	Бытовая техника	Сортировочн...	Дягтерев Д...	16.12...
16	Однокамерный холодильник ARDESTO DF - 90X	повре...	Бытовая техника	Сортировочн...	Дягтерев Д...	16.12...
17	Встраиваемый холодильник BEKO B 1751	новый	Бытовая техника	Сортировочн...	Дягтерев Д...	16.12...
18	Ноутбук HP ProBook 430 G5(3DP19ES) Silver	новый	Компьютерная тех...	База 1	Каргаполов ...	16.12...
19	Компьютер Dell Vostro 3470 SFF(N203VD3470_UBU)	б/у	Компьютерная тех...	База 1	Дягтерев Д...	16.12...
20	Компьютер Everest Home & Office 1020(1020_8429)	новый	Компьютерная тех...	База 1	Каргаполов ...	16.12...
21	Компьютер Dell OptiPlex 3060 MFF(N01603060MFF_U)	новый	Компьютерная тех...	База 1	Петрова Ви...	16.12...

Постачальники

Менеджери складу

Опції складу

Додати товар

Вибрати товар

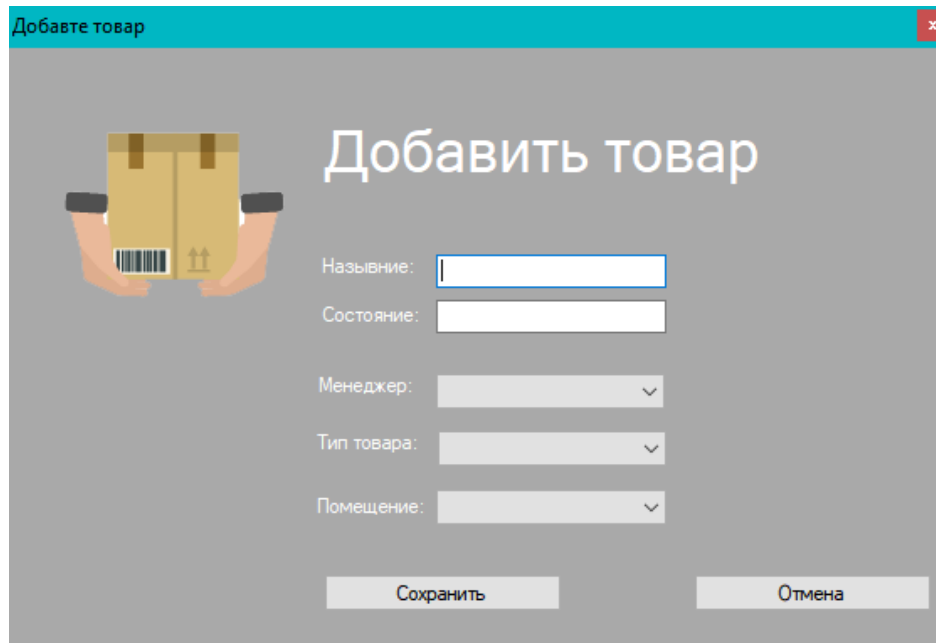
Вихід

Очистити базу

Відновити базу

Рисунок 5.2 – Головне вікно

На рисунку 5.3 та 5.4 користувач може редагувати список менеджерів складу та клієнтів. Можна додати нового менеджера або клієнта, можна вибрати одного працівника та вносити правки. Усі зміни йдуть до Бази даних.



Добавте товар

Добавить товар

Название:

Состояние:

Менеджер:

Тип товара:

Помещение:

Сохранить Отмена


Рисунок 5.5 – Додання нового товару

Тут ми бачимо процес переміщення товару. При переміщенні усі данні вносяться до бази даних, а вона, в свою чергу, додає нові данні до інших додатків системи. За допомогою цього вікна можна зробити погук товару по всій системі.

На рисунку 5.6 відображення інформації про обраний товар. Усі зміни вносилися до бази даних, тому користувач може переглянути усю історію переміщень товару по усім складам підприємства. Менеджер який працює х цією системою, може за допомогою цього вікна відправити товар клієнту або внести інші зміни, наприклад: змінити стан товару на «в процесі переміщення», стан «Знаходиться у приміщенні 123».

О выбранном товаре

Информация о товаре



Название товара: Газовая плита GORENJE GN
 Состояние: Отправлен клиенту ТОВ Инт
 Тип товара: Бытовая техника
 Помещение: База 1
 Ответственный: Дягтерев Дмитрий
 Дата изменения: 16.12.2018 21:53:37

Отправить клиенту
 Изменить состояние
 Переместить
 Назад

Транзакции

ID	Действие	Имя менеджера	Дата изменения
17	Товар Газовая плита GORENJE GN 5112 WF добавлен	Дягтерев Дмитрий	16.12.2018 21:51:08
26	Состояние товара "Газовая плита GORENJE GN 5112 WF" изменено с "новая" на "б/у"	Дягтерев Дмитрий	16.12.2018 21:53:03
27	Товар Газовая плита GORENJE GN 5112 WF перемещен из "Сортировочный склад" в "База 1"	Дягтерев Дмитрий	16.12.2018 21:53:09
28	Товар Газовая плита GORENJE GN 5112 WF отправлен клиенту "ТОВ Интели сенс"	Дягтерев Дмитрий	16.12.2018 21:53:37

Рисунок 5.6 – Інформація про товар

5.3 Проективання бази даних

Система бази даних є частиною практично будь-якої системи. Для розробки цих систем необхідно мати достатньо знань при розробці баз даних. Під час розробки програмного забезпечення, система баз даних, як правило, також інтегрується. У ідеальному випадку існуюча база даних може бути використана і може бути реалізоване лише з'єднання з нею. З іншого боку, інколи створюють свою приватну базу даних. В обох випадках знання в проектуванні баз даних необхідні.

Створення бази даних не вдається в один крок. Для цього необхідно пройти декілька фаз:

Зовнішня фаза: визначення інформаційної структури: здійснюється шляхом опису даних. Для цього потрібно визначити та структурувати інформаційні вимоги користувачів. Результат цього першого кроку, який також називається аналізом специфікації та вимог, є неформальним описом технічної проблеми.

Концептуальна фаза: Встановлення семантичної моделі: Метою концептуального дизайну є формалізований опис розглянутих фактів. Існує кілька

підходів до створення такої загальної картини. Найвідомішою моделлю є так звана модель відносин entity (ER model). У таких діаграмах для зображення сутності використовують прямокутники із описаними всередині полями сутності. Зв'язки між сутностями зображуються за допомогою ліній, та кількісної позначки біля відповідної сутності. ER-діаграму нашої системи спроектованої бази даних зображено на Рисунку 5.7

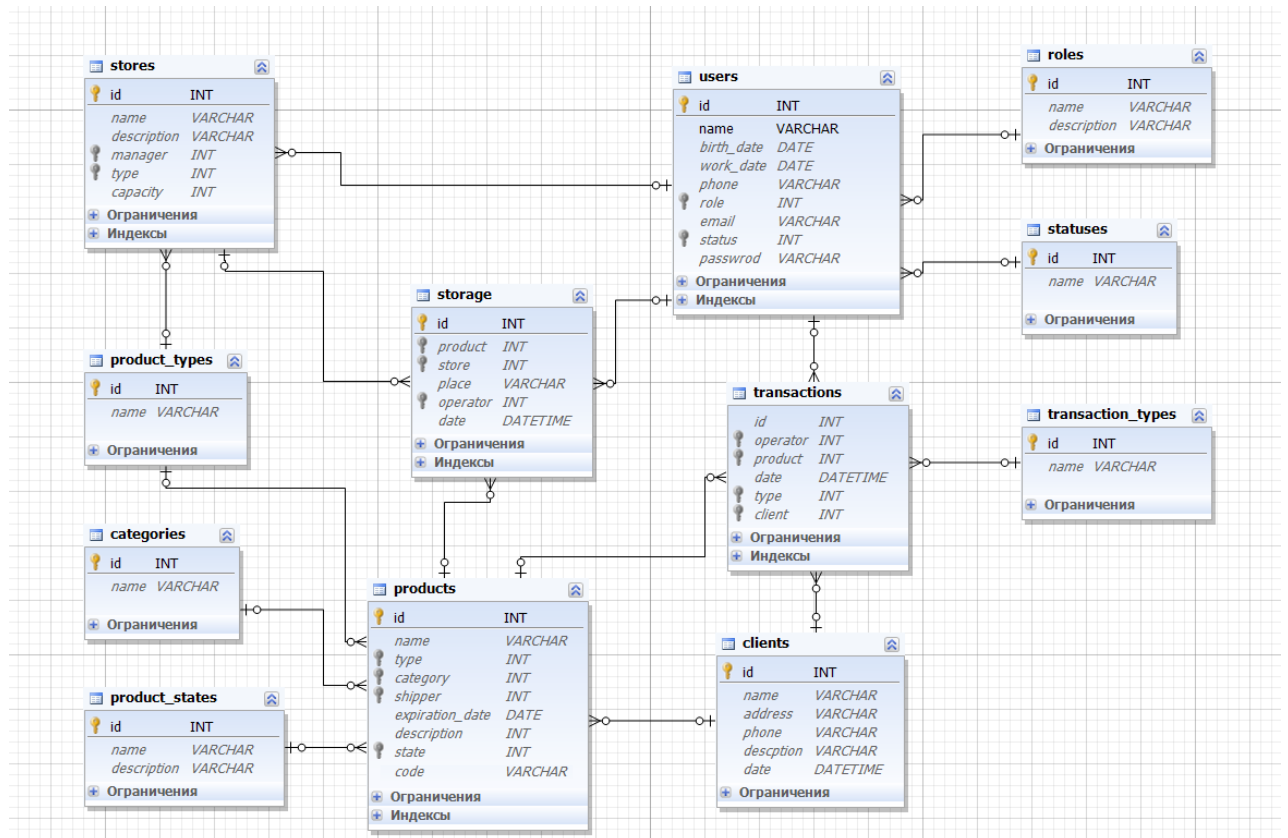


Рисунок 5.7 – ER діаграма бази даних системи

Логічний етап: Створення логічної моделі даних. Метою є передача моделі семантичних даних у логічну модель даних, наприклад. У реляційній моделі даних. Ця фаза складається з двох етапів: на першому етапі концептуальна схема (ER модель) повинна бути перетворена в схему бази даних. Цей крок можна автоматизувати за допомогою програмного забезпечення. На другому кроці оптимізація реляційної схеми.

Фізична фаза: Впровадження бази даних: наприкінці цієї фази порожня база даних повинна бути заповнена. Для цього логічна модель повинна бути переведена

в конкретну схему бази даних за допомогою мови визначення даного (наприклад, SQL). Ви повинні визначити типи даних, діапазони значень, відносини та представлення даних.

MySQL Workbench – це графічний інтерфейс користувача і набір інструментів для роботи з базами даних MySQL. Система може використовуватися для проектування, редагування, чіткого відображення і адміністрування баз даних. В MySQL Workbench являє собою графічний інструмент для моделювання і система розробки для баз даних MySQL. Він надає широкі функціональні можливості для повсякденної роботи з базами даних і може використовуватися для проектування, створення, редагування, адміністрування та відображення баз даних. Програмне забезпечення здатне витягувати структури з існуючих баз даних і робити їх зрозумілими.

MySQL Workbench доступний у безкоштовній і комерційній версії. Розробники можуть візуально проектувати бази даних в автономному режимі і розгортати їх на сервері MySQL. Для досвідчених користувачів, яким потрібні додаткові функції, в інструмент можуть бути вбудовані розширення з мовами сценаріїв. MySQL Workbench можна використовувати на комп'ютерах з операційними системами Linux, macOS або Microsoft Windows.

MySQL Workbench надає безліч функцій для роботи з базами даних MySQL і пропонує безліч способів їх використання. Центральні функції інструменту:

- Проектування і моделювання баз даних
- Розробляти, будувати і оптимізувати бази даних
- Адміністрування баз даних
- Документація баз даних
- Моніторинг та оптимізація продуктивності баз даних
- Міграція баз даних
- Зворотний інжиніринг баз даних

Оскільки MySQL Workbench має графічний користувацький інтерфейс, інструмент працює інтуїтивно і чітко навіть з дуже складними базами даних. Бази даних можуть візуально проектуватися, моделюватися, створюватися і

управлятися. Навіть складні налаштування можливі при відносно невеликих зусиллях і без спеціальних знань в області програмування. Інтегрований редактор SQL працює з кольоровою підсвіткою синтаксису і автозаповненням. Через графічну панель продуктивності, дозволяє переглядати і оптимізувати продуктивність баз даних. Звіти про продуктивність дозволяють легко ідентифікувати обчислювальні SQL-запити і операції. Ще однією важливою особливістю MySQL Workbench є підтримка міграції баз даних. Дані з серверів Microsoft SQL, Microsoft Access, Sybase ASE, PostgreSQL та інших відповідних систем управління базами даних легко переносити і перетворювати в MySQL. Використовуючи реверс-інжиніринг, моделі EER (розширені моделі взаємозв'язків сутностей) можна отримати з існуючих баз даних і створити заново.

Часто функції, які використовуються адміністраторами бази даних, виконують резервне копіювання і відновлення даних з бази даних MySQL. MySQL Workbench надає необхідні функції для цього і дозволяє виконувати резервне копіювання і відновлення всього декількома клацаннями миші.

Нижче наведено таблиці 5.1 - 5.12, де представлено опис полів WMS системи, “Тип товару”, “Категорія товару”, “Стан товару”, “Зберігання”, “Товар”, “Користувач”, “Дія (Транзакція)”, “Клієнт”, “Роль”, “Статус користувача”, “Тип дії” відповідно.

Назва	Тип
ID	integer
Назва	varchar
Опис	varchar
Менеджер	integer
Тип товарів	integer
Місткість	integer

Таблиця 5.1 – Опис полів таблиці “Складське приміщення”

Назва	Тип
ID	integer
Назва	varchar

Таблиця 5.2 – Опис полів таблиці “Тип товару”

Назва	Тип
ID	integer
Назва	varchar

Таблиця 5.3 – Опис полів таблиці “Категорія товару”

Назва	Тип
ID	integer
Назва	varchar
Опис	varchar

Таблиця 5.4 – Опис полів таблиці “Стан товару”

Назва	Тип
ID	integer
Товар	integer
Приміщення	integer
Місце	varchar
Оператор	integer
Дата прийому	datetime

Таблиця 5.5 – Опис полів таблиці “Зберігання”

Назва	Тип
ID	integer
Назва	varchar
Тип	integer
Категорія	integer
Постачальник	integer
Термін придатності	date
Опис	varchar
Стан	integer

Таблиця 5.6 – Опис полів таблиці “Товар”

Назва	Тип
ID	integer
ПІБ	varchar
Дата народження	date
Дата прийняття на роботу	date
Телефон	varchar
Роль	integer
email	varchar
Статус	integer

Таблиця 5.7 – Опис полів таблиці “Користувач”

Назва	Тип
ID	integer
Оператор	integer
Товар	integer
Дата дії	datetime
Тип дії	integer
Клієнт	integer

Таблиця 5.8 – Опис полів таблиці “Дія(Транзакція)”

Назва	Тип
ID	integer
Назва організації	varchar
Адреса	vachar
Телефон	varchar
Опис	varchar
Дата реєстрації	datetime

Таблиця 5.9 – Опис полів таблиці “Клієнт”

Назва	Тип
ID	integer
Назва	varchar

Опис	varchar
------	---------

Таблиця 5.10 – Опис полів таблиці “Роль”

Назва	Тип
ID	integer
Назва	varchar

Таблиця 5.11 – Опис полів таблиці “Статус користувача”

Назва	Тип
ID	integer
Назва	varchar

Таблиця 5.12 – Опис полів таблиці “Тип дії”.

5.4 Реалізація API

Термін API є короткою формою «Application-Programming-Interface»[2]. З перекладу на українську мову, це означає «інтерфейсу до прикладного програмування». У розмовній мові, проте, API зазвичай називають програмним інтерфейсом.

Цей інтерфейс надає іншим програмам інструмент для підключення до програмної системи. Це дозволяє розробникам впливати на обладнання, таке як монітор або дані на жорсткому диску, не звертаючись до них безпосередньо. Операційна система служить інтерфейсом, який отримує запити від програм через надані бібліотеки і передає їх апаратного забезпечення.

Проте, термін «API» став актуальним в основному завдяки використанню веб-сервісів. Це дозволяє розробникам використовувати надані інтерфейси для динамічної інтеграції вмісту, наданого в їх власну програму. Таким чином, API-

інтерфейси служать для обміну і подальшої обробки даних і контенту між різними веб-сайтами, програмами та постачальниками контенту. Крім того, вони дозволяють третім сторонам отримувати доступ до раніше закритих пулів даних і групам користувачів.

У більш технічній логіці API-інтерфейси схожі на машину, еквівалентну призначеному для користувача інтерфейсу, який був оптимізований для людей і, отже, «читабельний». API представляє собою програмно-орієнтований інтерфейс, тобто машинозчитуваний. Інтерфейс прикладного програмування забезпечує зрозумілий і структурований доступ до функцій серверної частини. Крім того, дані можуть обмінюватися, наприклад, в особливо простий для обробки і скороченій формі. На Рисунку 5.8 зображено API сервер в розробленій мною системі.

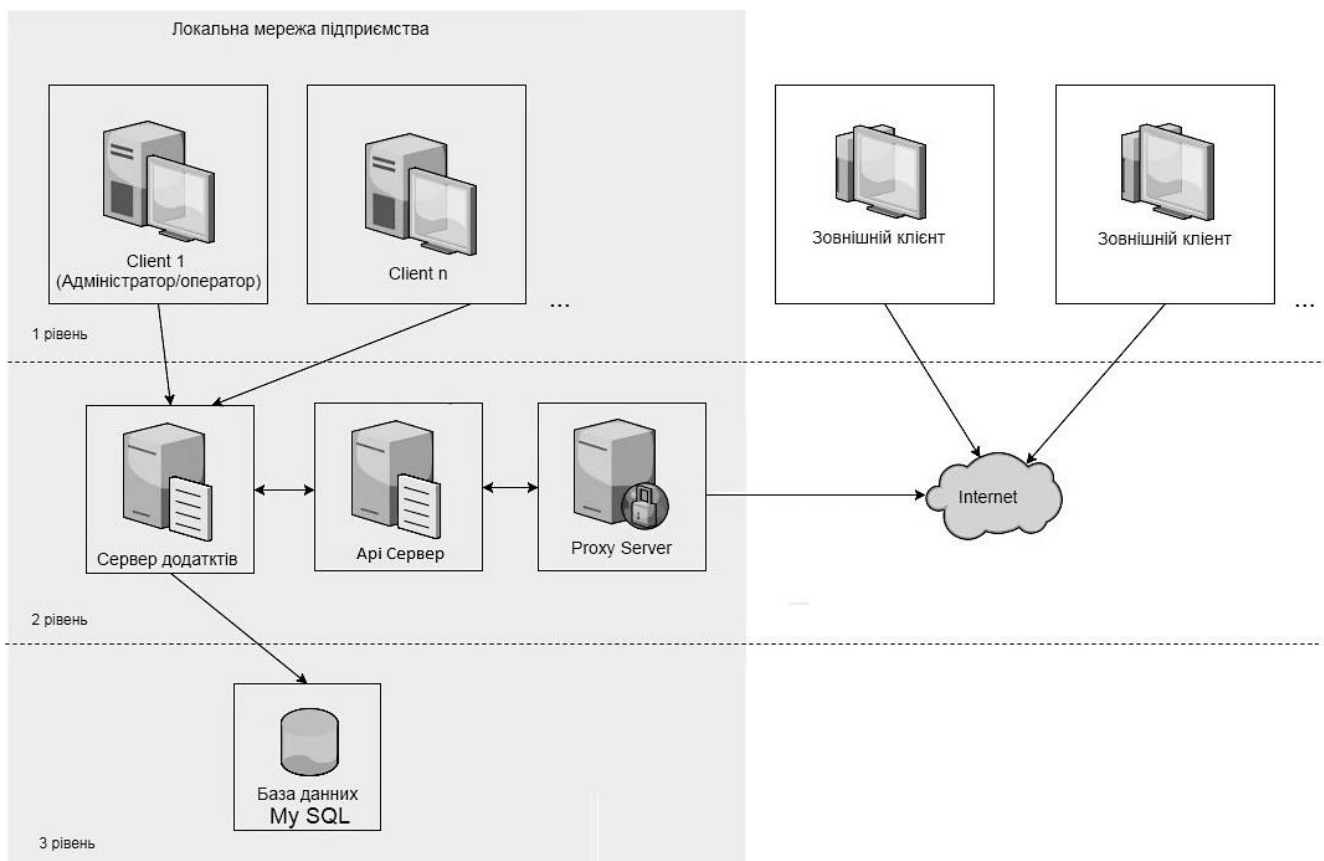


Рисунок 5.8 – API сервер в Локальній мережі підприємства

При програмуванні API об'єднують дані, що передаються між частинами програми, такими як модулі та програми.

Складні програми більше не працюють без API. У цьому випадку окремі частини програми, так звані модулі, вміщені реальним кодом. Модулі спілкуються виключно через API.

Простіше кажучи, API переводять програмне забезпечення на машинну мову, роблячи його читаним для різних компонентів. Завдяки програмним модулям і пов'язаним API-інтерфейсів складні програми легше обслуговувати. За допомогою переданих даних API перевіряє, чи працює модуль належним чином. Це полегшує пошук і виправлення помилок. Крім того, API-інтерфейси виступають в якості постачальників даних в області програмного забезпечення: контент може передаватися між різними веб-сайтами і програмами з використанням API-інтерфейсів. На Рисунку 5.9 зображено частинку коду, за допомогою якого було реалізовано API.

```

class Program
{
    private static Controller ctr = Controller.Instance;
    private static string errorResponseString = "{\"success\":false,\"error\":\"invalid query\"}";
    private static string html = "";
    static void Main(string[] args)
    {
        // = new ApiHttpServer();

        try
        {
            html = System.IO.File.ReadAllText(@"C:\store\index.html");

            // Display the file contents to the console. Variable text is a string.

            string webfixServerHost = "http://localhost:9999";

            ApiHttpServer.HttpListenerRequestHandler handlerForGetRequest =
                new ApiHttpServer.HttpListenerRequestHandler(HttpGetCallback);

            ApiHttpServer server = new ApiHttpServer(new Uri(webfixServerHost));

            server.OnGetRequest += handlerForGetRequest;

            server.StartListen();
            Console.WriteLine("ApiServer start: " + webfixServerHost);

            Console.WriteLine("Press [Enter] to quit.");
            Console.ReadLine();

            server.OnGetRequest -= handlerForGetRequest;
            server.StopListen();
        }
        catch (Exception exception)
    }
}

```

Рисунок 5.9 – Ралізація API

Багато проектів в даний час надають або використовують API на основі JSON, зазвичай RESTful. Комбінація JSON і REST досить поширена, але все API

трохи відрізняються. Будь-яка команда, яка працює над реалізацією JSON, отримає різні результати з точки зору точної структури JSON, іменування кінцевих точок REST і так далі. В результаті практично кожна команда створює власний індивідуальний API. У наведених нижче Рисунках зображена частина коду написана для пробної передачі даних до API server, та відображення у браузері.

```
[
  {
    "id": 1,
    "name": "Товар Компьютерный стол Максим Мах Комп 803 Matte добавлен",
    "item_id": 1,
    "user_id": 1,
    "date": "16.12.2018 21:51:07"
  },
  {
    "id": 2,
    "name": "Товар Стол Rondi Пилот-1 Ольга добавлен",
    "item_id": 2,
    "user_id": 2,
    "date": "16.12.2018 21:51:07"
  },
  {
    "id": 3,
    "name": "Товар Компьютерный стол + тумба Star - М Джуниор добавлен",
    "item_id": 3,
    "user_id": 2,
    "date": "16.12.2018 21:51:07"
  },
  {
    "id": 4,
    "name": "Товар Обеденный стол Vetro Mebel TM-51 - 1 Белый добавлен",
    "item_id": 4,
    "user_id": 1,
    "date": "16.12.2018 21:51:07"
  },
  {
    "id": 5,
    "name": "Товар Стол Rondi Кухня-2 Дуб Сонома добавлен",
    "item_id": 5,
    "user_id": 2,
    "date": "16.12.2018 21:51:07"
  },
  {
    "id": 6,
    "name": "Товар Обеденный стол МИКС - Мебель Говерла 160x90x75 см Темный орех добавлен",
    "item_id": 5,
    "user_id": 1,
    "date": "16.12.2018 21:51:07"
  }
],
```

Рисунок 5.10– Реалізація API

```
{
  "id":1,"name":"Товар Компьютерный стол MaximM Maxi Комп 803 Matte добавлен","item_id":1,"user_id":1,"date":"16.12.2018 21:51:07"},
  {"id":2,"name":"Товар Стол Rondi Пилот-1 Ольга добавлен","item_id":2,"user_id":2,"date":"16.12.2018 21:51:07"},
  {"id":3,"name":"Товар Компьютерный стол + тумба Star - М Джуниор добавлен","item_id":3,"user_id":2,"date":"16.12.2018 21:51:07"},
  {"id":4,"name":"Товар Обеденный стол Vetro Mebel TM-51 - 1 Белый добавлен","item_id":4,"user_id":1,"date":"16.12.2018 21:51:07"},
  {"id":5,"name":"Товар Стол Rondi Кухня-2 Дуб Сонома добавлен","item_id":5,"user_id":2,"date":"16.12.2018 21:51:07"},
  {"id":6,"name":"Товар Обеденный стол МИКС - Мебель Говерла 160x90x75 см Темный орех добавлен","item_id":5,"user_id":1,"date":"16.12.2018 21:51:07"},
  {"id":7,"name":"Товар Стул Примтекс Плюс 1011 alum S-64 Светло - бежевый добавлен","item_id":6,"user_id":2,"date":"16.12.2018 21:51:07"},
  {"id":8,"name":"Товар Стул Vetro Mebel M-11 Гаванна(M - 11 - gavana) добавлен","item_id":7,"user_id":2,"date":"16.12.2018 21:51:07"},
  {"id":9,"name":"Товар Стул Special4You Sedia White(E5746) добавлен","item_id":8,"user_id":1,"date":"16.12.2018 21:51:07"},
  {"id":10,"name":"Товар Стул Vetro Mebel M - 21 Синий(M - 21 - blu) добавлен","item_id":9,"user_id":1,"date":"16.12.2018 21:51:07"},
  {"id":11,"name":"Товар Табурет Новый Стиль Teddy Chrome V-19 добавлен","item_id":10,"user_id":2,"date":"16.12.2018 21:51:07"},
  {"id":12,"name":"Товар Стиральная машина с двумя барабанами LG FH6G1BCH2N / FH8G1MINI3 добавлен","item_id":10,"user_id":3,"date":"16.12.2018 21:51:07"},
  {"id":13,"name":"Товар Стиральная машина полногабаритная LG FH0B8QD добавлен","item_id":11,"user_id":3,"date":"16.12.2018 21:51:07"},
  {"id":14,"name":"Товар Стиральная машина полногабаритная BEKO WMY 81233 LMB3 добавлен","item_id":11,"user_id":3,"date":"16.12.2018 21:51:08"},
  {"id":15,"name":"Товар Газовая плита HANSA FCGW52028 добавлен","item_id":12,"user_id":3,"date":"16.12.2018 21:51:08"},
  {"id":16,"name":"Товар Газовая плита GEFEST 3200 - 06 K19 добавлен","item_id":13,"user_id":3,"date":"16.12.2018 21:51:08"},
  {"id":17,"name":"Товар Газовая плита GORENJE GN 5112 WF добавлен","item_id":14,"user_id":3,"date":"16.12.2018 21:51:08"},
  {"id":18,"name":"Товар Однокамерный холодильник VESTFROST VD142RS добавлен","item_id":15,"user_id":3,"date":"16.12.2018 21:51:08"},
  {"id":19,"name":"Товар Однокамерный холодильник ARDESTO DF - 90X добавлен","item_id":16,"user_id":3,"date":"16.12.2018 21:51:08"},
  {"id":20,"name":"Товар Встраиваемый холодильник BEKO B 1751 добавлен","item_id":17,"user_id":3,"date":"16.12.2018 21:51:08"},
  {"id":21,"name":"Товар Ноутбук ASUS X570UD - E4037(90NB0HS1 - M00460) Black добавлен","item_id":17,"user_id":3,"date":"16.12.2018 21:51:08"},
  {"id":22,"name":"Товар Ноутбук HP ProBook 430 G5(3DP19ES) Silver добавлен","item_id":18,"user_id":2,"date":"16.12.2018 21:51:08"},
  {"id":23,"name":"Товар Компьютер Dell Vostro 3470 SFF(N203VD3470_UBU) добавлен","item_id":19,"user_id":1,"date":"16.12.2018 21:51:08"},
  {"id":24,"name":"Товар Компьютер Everest Home & Office 1020(1020_8429) добавлен","item_id":20,"user_id":2,"date":"16.12.2018 21:51:08"},
  {"id":25,"name":"Товар Компьютер Dell OptiPlex 3060 MFF(N01603060MFF_U) добавлен","item_id":21,"user_id":4,"date":"16.12.2018 21:51:08"},
  {"id":26,"name":"Состояние товара \"Газовая плита GORENJE GN 5112 WF\" изменино с \"Новая\" на \"6/у\"","item_id":14,"user_id":3,"date":"16.12.2018 21:53:03"},
  {"id":27,"name":"Товар Газовая плита GORENJE GN 5112 WF перемещен из \"Сортировочный склад\" в \"База 1\"","item_id":14,"user_id":3,"date":"16.12.2018 21:53:09"},
  {"id":28,"name":"Товар Газовая плита GORENJE GN 5112 WF отправлен клиенту \"ТОВ Интели сенс\"","item_id":14,"user_id":3,"date":"16.12.2018 21:53:37"}
}
```

Рисунок 5.11 – кодування інформації JSON

5.5 Реалізація програмного додатку

Розробка додатку для Android та Apple було реалізовано за допомогою кількох платформ та з однаковим інтерфейсом. Це доповнення являє собою систему, яка демонструє мобільний додаток, який спрощує реалізацію системи на телефоні або на планшеті. Ми розглянемо версію для iPhone.

Програмний продукт використовує Cocos2d для налаштування сторонніх залежностей для роботи з кодом на MacBook або іншій Mac OS платформі. Використовується також Realm Mobile Platform. Realm - це база даних для декількох платформ[8]. Realm Mobile Platform - це шар даних для наступного покоління для додатків. Якщо коротко, то це нативна «без SQL» база даних для Android (Java, Kotlin), iOS (Objective-C, Swift), Xamarin (C#) і JavaScript (React Native, Node.js). Так само є бекенд, який дозволяє синхронізувати дані з усіх джерел.

Для роботи з ним, також потрібний робочий примірник Realm Object Server, для того, щоб система могла виконувати завдання та переносити данні між екземплярами. Також для роботи з програмним кодом, Realm може взаємодіяти з

Mac OS та для Linux і Ubuntu. Якщо потрібно працювати з Linux, то потрібні сторонні налаштування для доступу до сервера Linux.

Для роботи Додатку, використовується сервер Realm Object Server. Працювати та вносити зміну у програмний код продукту, може лише користувач з правами адміністратора на Object Server. Далі, для компілювання використовуємо Cocoapods та запускаємо додаток через XCode або Rider для Android. На рисунку 5.12 Зображено Use case діаграму взаємодії додатку з робочою станцією. Розглянемо її.

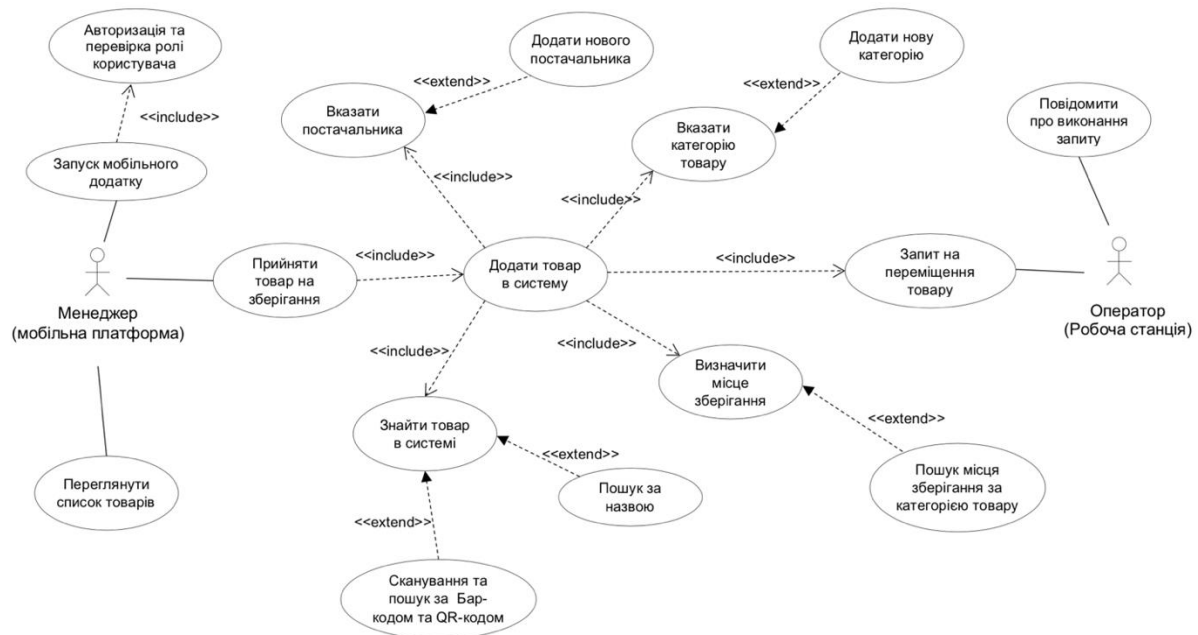


Рисунок 5.12 – Use Case діаграма системи

Данна система підтримує ряд простих функцій. Розглянемо їх.

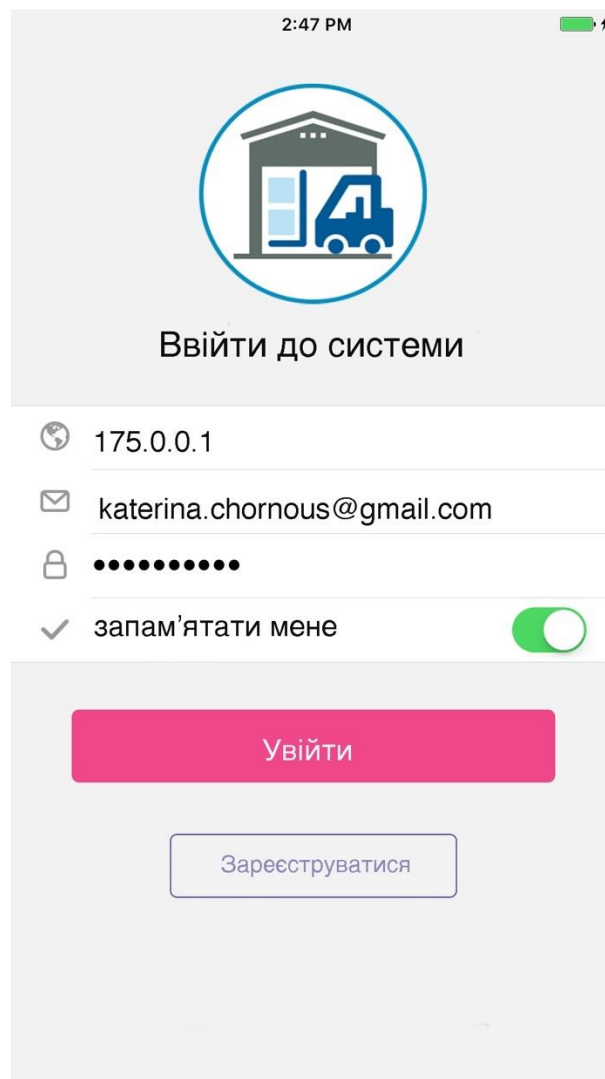


Рисунок 5.13 – Вхід до системи

Вхід до системи. В систему можна ввійти як зареєстрований користувач, або зареєструватися використовуючи пошту та сервер (Рисунок 5.13).

Навігація по системі. Простий інтерфейс з підтримкою «Tab Bar» та стандартними налаштуваннями як зображено на Рисунку 5.14. Також, через налаштування можна внести зміни у профіль (Рисунок 5.15).

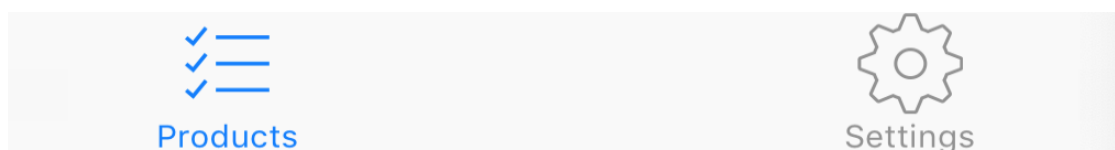


Рисунок 5.14 – Панель налаштувань

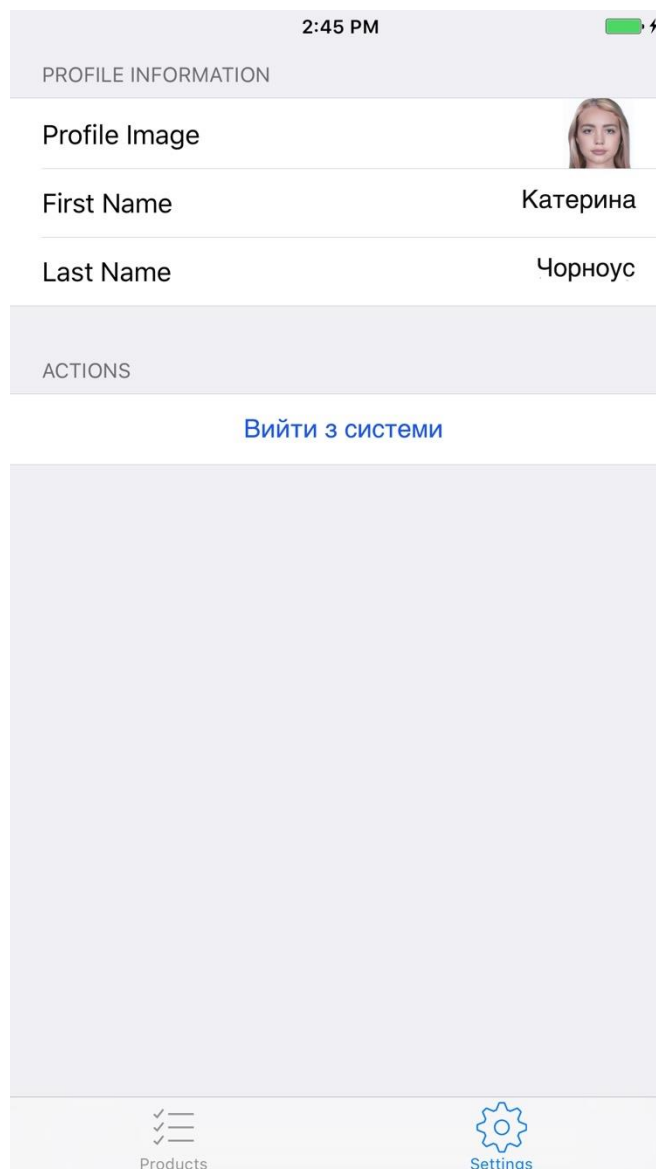


Рисунок 5.15 – Профіль користувача

Панель вкладок. Логіка додатку дуже проста, вона синхронізується з десктопною версією, та підтягує з бази даних всі файли та данні та вносить у основний список з продуктами.

Одними з плюсів додатку, є можливість одночасно користуватися їм кількома користувачами. Щоб налаштувати додаток для багатокористувацького режиму, потрібно першому користувачу-адміністратору надати на сервері можливість доступу іншим користувачам системи.

Списки товарів. Коли система підтягнула данні з бази, можна переглянути усі наявні позиції (Рисунок 5.16) Також їх можна фільтрувати, натискаючи у верхньому лівому куту параметр сортування.

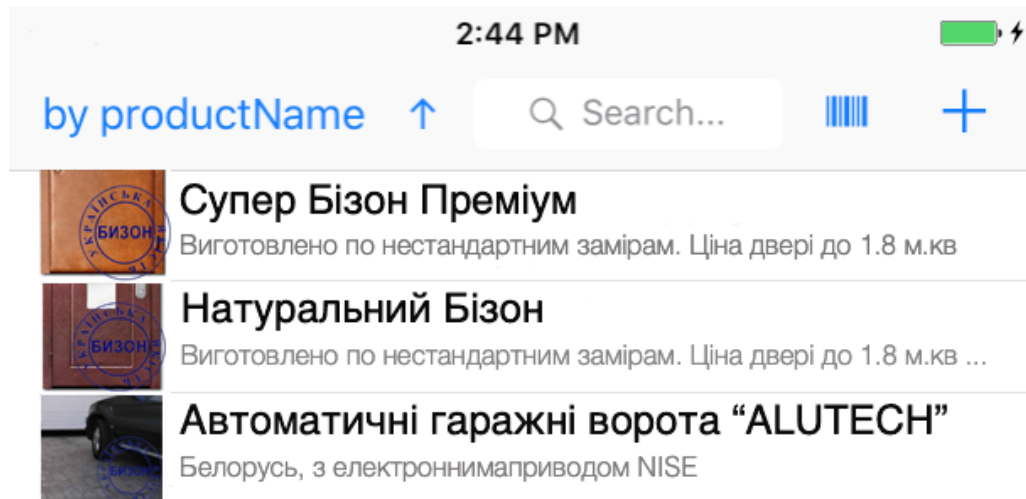


Рисунок 5.16 – Список товарів

Сканування Бар-коду та QR-коду. Для використання цього інструменту, користувач повинен відкрити доступ до камери смартфона. (Рисунок 5.17).

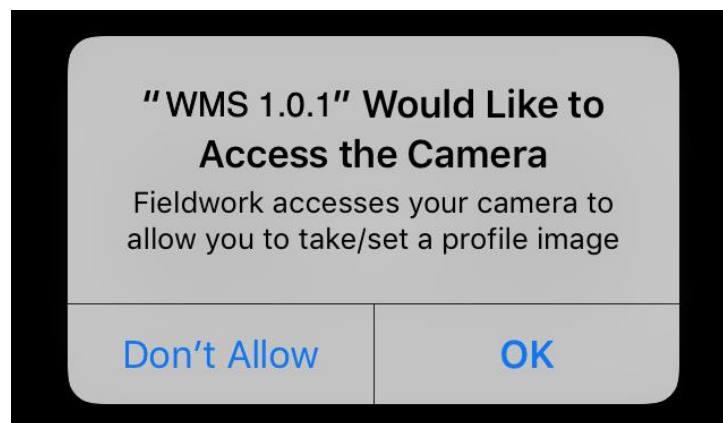


Рисунок 5.17 – Доступ для сканування

Коли доступ відкрито, користувач повинен помістити бар код до прямокутника (Рисунок 5.18), та система автоматично відкриє наступне вікно.

3:57 PM

68%

Close Скандування Бар-коду



Рисунок 5.18 – Скандування Бар Коду

Додавання в ручну. Якщо товар не з'являється, або програма не може розпізнати бар код, користувач повинен внести новий товар вручну. Він може ввести ID товару, назвати товар, додати фото товару, Бар код та описання товару. Готовий, від сканований та доданий товар в системі виглядає так як на Рисунку 5.19.

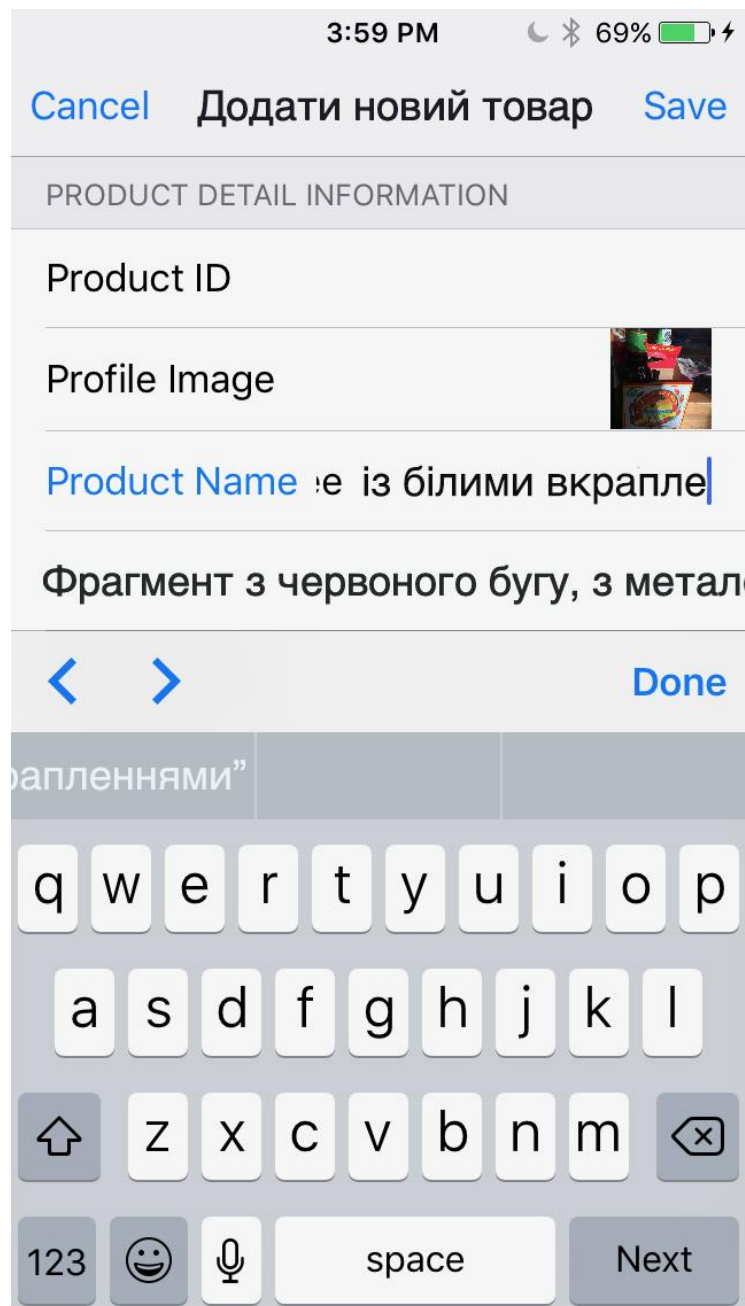


Рисунок 5.19 – Відсканований товар

Основний екран також підтримує можливість пошуку товару, яка підтримує пошук по всім елементам назви товару. Користувач системи також може видалити товар.

Система буда написана за допомогою двох основних класів: Клас Product (Рисунок 5.20) та класу Transaction (Рисунок 5.21). Product клас, реалізує те, як просто буде виглядати продукт у системі підприємства. Клас Transactions відстежує переміщення товару по системи. Він шукає інформацію про товар,

про додання нового товару або видалення; про додання нового бар коду, про додання товару до Бази даних. Також я використовую властивості, що б мати змогу відшукати усі транзакції на двох системах. Також, як і в десктопній версії, є можливість переглянути усі транзакції зареєстровані системою.

```
class Product : Object {
    dynamic var id = ""
    dynamic var creationDate: Date?
    dynamic var lastUpdated: Date?
    dynamic var productName = ""
    dynamic var productDescription = ""
    dynamic var image : Data?
    var amount: Int {
        get {
            return self.quantityOnHand()
        }
    }
    let transactions = List<Transaction>()
```

Рисунок 5.20 – Клас Product

```
class Transaction : Object {
    dynamic var id = NSUUID().uuidString
    dynamic var transactionDate: Date?
    dynamic var transactedBy = ""
    dynamic var productId = ""
    dynamic var amount = 0

    override static func primaryKey() -> String? {
        return "id"
    }
}
```

Рисунок 5.21 – Клас Transaction

6 ТЕСТУВАННЯ СИСТЕМИ

Тестування програмного забезпечення є важливим інструментом забезпечення якості систем і продуктів. Часто вони виявляють небажану поведінку програмного забезпечення. Тести програмного забезпечення є визнаними областями розробки програмного забезпечення. По суті, тестування програмного забезпечення може використовуватися як для валідації користувачів і клієнтів, так і для верифікації відповідно до цієї специфікації.

Під час тестового проекту виникають наступні завдання:

Управління тестуванням: управління проектом або управління підпроектом, концепція тестування, планування тестування і контроль

Аналіз випробувань: координація та підтримка відділу в створенні проектів випробувань, контроль виконання випробувань і контроль планів випробувань

Тестування: планування і виконання конкретних тестів (інтерфейси, міграція, продуктивність і т. Д.)

Автоматизація тестування: створення і підтримка автоматизованих процедур тестування

Підтримка тестування: розгортання тестових середовищ, тестування систем, запуск автоматичних тестів

Наступні типи тестів найбільш часто використовуються:

Модульні тести (тести для забезпечення цілісності окремих компонентів системи і їх інтерфейсів), системні тести, інтеграційні тести (тести для перевірки повноти необхідних функціональних вимог.) На підставі цих тестів регресивні тести можуть виконуватися автоматично.), тести на навантаження і продуктивність (автоматичні тести, які можна синхронізувати з поточним станом розробки.)
Наслідки зміни системи швидко виявляються і можуть бути негайно усунуті.)

6.1 Модульні тести

Модульні тести (Unit Tests) перевіряють, щоб компоненти, які написані розробниками, працюють як задумано. У гнучких методах забезпечення якості

вимагає дуже частого виконання тестів компонентів. Це може бути досягнуто тільки в тому випадку, якщо тести повністю автоматизовані, тобто вони самі є програмою, виконання якої вимагає не більше зусиль, ніж натискання кнопки. Зазвичай (але не обов'язково) тест пишеться тією ж мовою, що і об'єкт тесту. Гнучка розробка програмного забезпечення широко використовує тестування компонентів як частина керованої тестуванням розробки і рефакторинга[12].

Роль модульного тестування в гнучкою розробці програмного забезпечення значно відрізняється від ролі, яку вони грають в традиційних підходах. Таким чином, під час їх створення в контексті розробки, заснованої на тестуванні, вони більше служать для контролю розробки програмного забезпечення, ніж для виявлення дефектів. В контексті поетапного проектування вони необхідні для збереження навіть довготривалого змінюваного коду. Таким чином, вони є важливою передумовою для рефакторинга. Тести виявляють випадкові зміни в поведінці одним натисканням кнопки. Навіть якщо нові функції додаються в існуючу систему, модульні тести допомагають: виявляти ненавмисний вплив нових функцій на існуючі компоненти системи. Крім того, вони також допомагають документувати систему,

Було запропоновано невелику платформу, яка підтримує створення таких автоматичних тестів. Цей фреймворк SUnit можна вважати батьком цілого сімейства фреймворків для різних мов програмування. Представником цього сімейства для Java є середовище JUnit, яка була розроблена Кентом Беком і Еріхом Гамою.

Для проведення модульного тестування системи було використано пакет MSTest. У наступному прикладі (Рисунок 6.1) показаний результат проходження системою тестувань.

Test Name	Duration (мс)
addClientTest	279
AddItemTest	269
AddManagerTest	286
AddPlacementTest	275
AddTransactionTest	276
GetClientNameTest	266
GetClientsTest	273
GetItemByIdTest	671
GetItemNameTest	266
GetItemsTest	283
GetItemsTypeTest	257
GetPlacementNameTest	285
GetPlacementsTest	292
GetTypeNamesTest	286
GetUserNameTest	284
GetUsersTest	274
UpdateClientTest	293
UpdateItemTest	286
UpdateManagerTest	276

Сводка
 Последний тестовый запуск **Пройден** (Общее время выполнения 0:00:06,6494018)
 ✓ Тестов: 19 Пройден

Рисунок 6.1 – Тестування системи.

6.2 Інтеграційне тестування

Різні процедури тестування повинні допомогти відновити правильність коду: модульні тести використовуються для тестування компонентів незалежно один від одного і гарантують функціональну коректність. Інтеграційні тести потім гарантують, що взаємодія і зв'язок між компонентами безпомилкові.

Мета інтеграційного тестування полягає в тому, щоб виявити помилки, які не може знайти один модульний тест. Це відбувається головним чином тому, що модульне тестування розглядає окремі модулі складного програмного проекту незалежно від всіх інших. Це добре, якщо просто переконатися, що код виконує те, що ви очікуєте, але зовсім недостатньо, щоб забезпечити безперебійний зв'язок з іншими модулями.

І саме тут вступають інтеграційні тести: вони намагаються виявити помилки, що виникають в результаті взаємодії між окремими компонентами. Це може бути несумісність в використовуваних форматах повідомлень або даних або навіть помилки в самих даних - наприклад, неправильні введення і виведення, які раптово закривають функцію.

Інтеграційні тести стають абсолютно незамінними, коли ваші власні послуги залежать від зовнішніх систем. Типовим сценарієм є підключення до веб-додатків, таким як Facebook, Twitter & Co. або використання їх API, протоколів і форматів даних для їх власного веб-проекту.

Згодом на практиці були створені різні стратегії для проведення інтеграційних тестів, кожна з яких має свої переваги і, на жаль, також недоліки. Щоб пом'якшити останнє, процедури часто використовуються в зв'язці. Ось дві широко поширені стратегії інтеграції:

Зверху вниз: компоненти в вищих шарах абстракції спочатку тестуються, а нижче лежаче моделюються так званими фіктивними або фіктивними об'єктами. Особливо на початку, розробка цих наповнювачів є відносно дорогою. В якості нагороди, проте, ви швидко отримуєте першу виставу про систему в цілому і можете поступово стимулювати розробку відсутніх компонентів.

Знизу вгору: крок за кроком окремі компоненти, які вже були завершені, пов'язані і протестовані, так що вся система, нарешті, відображається. З фіктивними об'єктами дійсно повністю відмовилися, але також з'явилася можливість раннього моделювання всієї системи. Якщо при подальшій збірці деталей виникають проблеми, локалізація і виправлення помилок можуть стати досить складними.

Плюс інтеграційного тестування полягає не тільки в тому, що існуючі системи модульних тестів все ще можна використовувати і не потрібно створювати нові системи, але, перш за все, в простій розробці реальних тестів. Якщо ви не хочете вдаватися до допомоги спеціалізованого інженера-тестувальника, навіть програміст з досвідом розробки модульних тестів може взяти на себе ці завдання.

У той же час саме ця подібність між двома тестовими процедурами є серйозною проблемою: на практиці можна знову і знову спостерігати, що розробники дійсно хочуть створити модульний тест, але в підсумку отримують інтеграційний тест. У двох словах, можна навіть сказати, що будь-яка інтеграція зовнішніх систем або компонентів або взаємодія принаймні двох компонентів однієї і тієї ж системи вже є інтеграційний тест.

Але чому відмінність між двома тестами так важливо? Просто: за визначенням модульні тести повинні бути дуже швидкими, так як вони використовуються розробниками щодо часто і навіть для невеликих змін коду. Звичайно, інтеграційні тести набагато повільніші, тому що вони засновані на зовнішніх компонентах і не повинні виконуватися постійно.

Якщо зміни в коді час від часу призводять до збоїв модульних тестів - які насправді є інтеграційними тестами - і якщо ці тести також займають непотрібне кількість часу, вся система модульних тестів занадто рада, щоб їх відкинути. Чіткий поділ двох процедур випробувань може успішно протидіяти такому непотрібному відмови.

6.3 Системне тестування

В рамках тесту системи тестувальники повинні переконатися, що в програмному забезпеченні виявлені помилки, перш ніж воно буде доставлено користувачам або замовникам. Вони тестують не тільки програмне забезпечення, але і впроваджені бізнес-процеси і систему в цілому. Системний тест - дуже складний проект, що вимагає власних методів та інструментів.

На даному етапі, в програмних продуктах не було виявлено ніяких порушень, тестування проводилося, як перевірка даних, що повертаються з контролерів, приклад виводу інформації показано на рисунку 9.2


```

namespace Tests
{
    [TestClass]
    public class TestController
    {
        [TestInitialize]
        public void TestInit()
        {
            Controller ctr = Controller.TestInstance;
            ctr.clearBase();

            ctr.addManger("Иванов Василий",22, DateTime.Today.ToString());
            ctr.addManger("Каргаполов Юрий",31, DateTime.Today.ToString());
            ctr.addManger("Дягтерев Дмитрий",25, DateTime.Today.ToString());

            ctr.addClient("ЧП Прайм Тай", "051-123-213-33");
            ctr.addClient("ЗАТ Петровский кабинет", "051-723-612-54");
            ctr.addClient("ТОВ Интели сенс", "031-563-233-53");

            ctr.addPlacement("Главный склад",1);
            ctr.addPlacement("Сортировочный склад",2);
            ctr.addPlacement("База 1",1);
            ctr.addPlacement("База 2",1);

            ctr.addItem("Товар 1", "новый",2,3,1);
            ctr.addItem("Товар 2", "б/у",1,1,2);
            ctr.addItem("Товар 3", "новый",1,1,2);
            ctr.addItem("Товар 4", "новый",3,4,3);

            ctr.addItemType("Стол");
            ctr.addItemType("Шкаф");
            ctr.addItemType("Стул");
        }
        [TestCleanup]
        public void TestCleanup()
        {
            Controller ctr = Controller.TestInstance;
            ctr.clearBase();
        }
        [TestMethod]
        public void GetItemByIdTest()
        {
            Controller ctr = Controller.TestInstance;
            var item = ctr.getItemById(1);
            Assert.IsTrue(item.id == 1);
        }
        [TestMethod]
        public void AddItemTest()
        {
            Controller ctr = Controller.TestInstance;
            ctr.addItem("testName", "testState", 1, 1, 1);
            var items = ctr.getItems();

            bool isAdded = false;

            foreach (var it in items)
                if (it.name == "testName")
                    isAdded = true;

            Assert.IsTrue(isAdded);
        }
    }
}

```

Рисунок 6.2 – Приклад частити системного тестування

6.2 UI тести

Якщо ви подивіться на добре відому піраміду автоматизації тестування, автоматизація тестування користувальницького інтерфейсу відображається як найменша одиниця виміру:

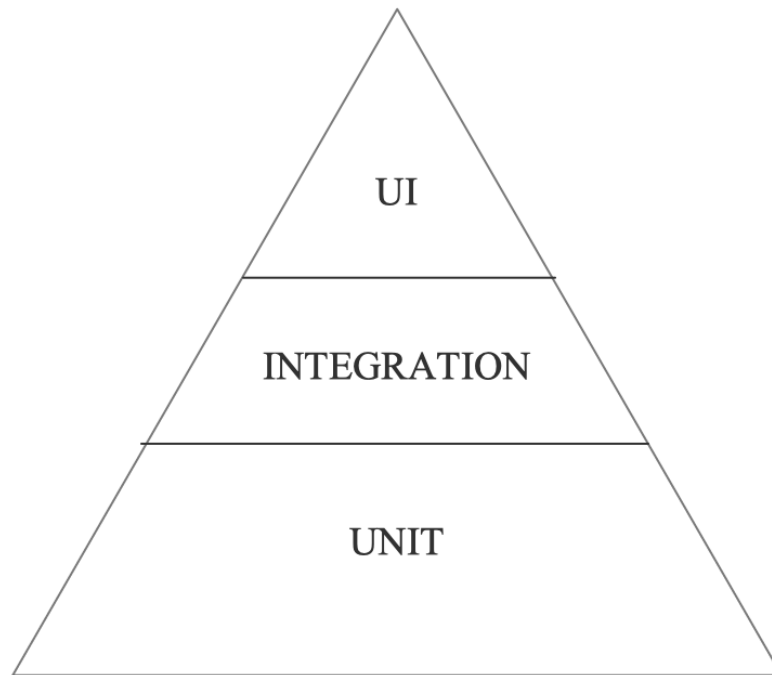


Рисунок 6.2 – Піраміда автоматизації тестування

В даний час існує безліч інструментів автоматизації тестування користувацького інтерфейсу. Незалежно від того, який інструмент ви використовуєте для автоматизації тестування користувацького інтерфейсу, інструменти фокусуються тільки на одному (технічному) аспекті автоматизації: знаходженні і зверненні до компонентів призначеного для користувача інтерфейсу. Для виконання дій над ними кожен інструмент пропонує інтерфейс. Для деяких інструментів цей інтерфейс є відносно великим, з багатьма діями для багатьох типів компонентів (таких як Jubula). Інші інструменти (такі як Selenium) мають дуже спрощений API, який може (або повинен) програмувати адресацію більш складних компонентів самостійно. Однак жоден інструмент не вказує, як професійно побудувати ці тести. Ось в чому проблема. API можна використовувати для автоматизації окремих кроків. Однак різниця між серією автоматизованих кроків і розумним, підтримуваним, аналізованим контрольним прикладом велика і значна.

За допомогою цих технічних API-інтерфейсів дуже легко автоматизувати ваші власні кроки робочого процесу (ми явно не називаємо їх «тестовими прикладами» в таких випадках). Це може статися все швидше з популярним підходом

відтворення захоплення. Тим не менш, це також дуже легко, якщо ви пишете тести вручну.

Друга проблема - це область застосування тестів для користувача інтерфейсу. Як згадувалося вище, є кілька рівнів тестування, які автоматично тестуються в проекті. Занадто часто інструмент автоматизації тестування користувальницького інтерфейсу використовується для тестів, які краще виконувати на більш низьких рівнях (наприклад, модульне тестування).

Для другого програмного додатку було написано кілька UI тестів. Приклад коду зображено на Рисунку 6.3

```
import XCTest

@testable import ImageRow
import Eureka

class ImageRowTests: XCTestCase {

    var formVC = FormViewController()

    var availableSources: ImageRowSourceTypes {
        var result: ImageRowSourceTypes = []

        if UIImagePickerController.isSourceTypeAvailable(.photoLibrary) {
            result.insert(ImageRowSourceTypes.PhotoLibrary)
        }
        if UIImagePickerController.isSourceTypeAvailable(.camera) {
            result.insert(ImageRowSourceTypes.Camera)
        }
        if UIImagePickerController.isSourceTypeAvailable(.savedPhotosAlbum) {
            result.insert(ImageRowSourceTypes.SavedPhotosAlbum)
        }
        return result
    }
}
```

Рисунок 6.3 – UI тестування

Представлені вище концепції повинні бути знайомі багатьом розробникам. Звичайно, якщо говорити про окремі рівні абстракції, інкапсуляції і згуртованості. Однак той факт, що концепції добре відомі, не означає, що вони використовуються успішно або правильно в автоматизації призначеного для користувача інтерфейсу. З технічної точки зору, якщо фахівці з тестування використовуються без

відповідної підготовки для автоматизації тестування користувальницького інтерфейсу, можуть виникнути конфліктуючі розділи. Незалежно від того, чи є ви тестувальником або розробником, кожен член команди повинен стежити за відповідною структурою автоматизованих тестів для користувача інтерфейсу і постійно розширювати свої знання.

6.5 Мануальне тестування

Автоматизоване тестування більше не є чимось новим. Кожне гарне програмне забезпечення розроблене за допомогою автоматичних модульних тестів і інтеграційних тестів, щоб завжди знати, чи все працює правильно. Є досить фреймворків і інструментів звітності. Вони знайшли своє місце на ринку і також майже незамінні. Той, хто говорить, що ручне тестування – річ минулого, неправий. Потрібні технічні знання, які перевіряють як концепцію, так і зручність використання, можливо, виявляючи помилки програмування (як при реалізації логіки, так і при виконанні тестів). І це вимагає когось, хто просто «стрибає», тому що завжди будуть нові користувачі, для яких програмне забезпечення є абсолютно новим і які точно так надходять. Частина створених тестів для розробленої системи, зображено у таблицях 6.1 – 6.2

Таблиця 6.1 – Тестовий сценарій входу у мобільний програмний додаток.

Назва	Тест реєстрації	
Дія користувача	Очікуваний результат	Результат тесту
Попередні умови		
Відкрийте програмний додаток	Програмний додаток відкрито	Пройдений
Опис тестового сценарію		

Відкрити входу	Форму з полями вводу даних для реєстрації відкрито	Пройдений
Заповнити текстові поля значеннями: «IP»: 175.0.0.1 «Пошта»: Katerina.chornous@gmail.com «Пароль»: 1234567	Поля заповнені даними.	Пройдений
Натисніть «Увійти»	Відсутні повідомлення про помилку.	Пройдений
	Перехід на сторінку «Введіть дані правильно»	Пройдений

Таблиця 6.2 – Тестовий сценарій використання вікна редагування менеджерів.

Назва	Тест виконання замовлення готового товару	
Дія користувача	Очікуваний результат	Результат тесту
Попередні умови		
Відкрийте програмний додаток	Програмний додаток відкрито	Пройдений

Виконати аутентифікацію	Аутентифікація успішна, переадресовано на сторінку із переліком товарів	Пройдений
-------------------------	---	-----------

Продовження таблиці 9.3

Назва	Тест виконання замовлення готового товару	
Дія користувача	Очікуваний результат	Результат тесту
Опис тестового сценарію		
Натиснути на «Менеджери складу»	Здійснений перехід на сторінку зі списком менеджерів складу. На екрані відображається ІД працівника, ім'я, вік, Дата прийняття на роботу та клавіші «Додати», «Редагувати», «Назад»	Пройдений
Клацнути на кнопку «Редагувати»	На екрані відображається вікно з надписом «Редагування». В ньому є можливість редагування ім'я менеджера, віку, дати зарахування на роботу та кнопки «Зберегти» та «Відміна»	Пройдений

Змінити ім'я менеджера в вікні «Ім'я»	Здійснюється редагування Імені	Пройдений
Клацнути кнопку «Зберегти»	Здійснений перехід до вікна з формою, де відображена вже оновлена інформація про обраного нами працівника	Пройдений

7 СТАРТАП ПРОЕКТ

Кожен, хто хоче створити свій власний проект, починає як власник стартап-проекту. Ви придумали продукт або послугу, і хотіли б зробити щось краще, ніж те, що представлено зараз на ринку. Що з чого потрібно почати, щоб проект не зупинився на етапі створення та релізу? Я написала декілька найважливіших пунктів на етапі створення проекту та виведення на ринок:

1. Переконлива бізнес-ідея;
2. Стати відомим завдяки правильній рекламі;
3. Створити бізнес план;
4. Знайти фінансування для проекту.

Жодна компанія не може існувати дуже довго, жоден проект не може успішно працювати, якщо бізнес-ідея не правильно представлена майбутнім клієнтам. Це може звучати жорстоко, але реальність доводить це знову і знову. Досвід інших підприємців допоміг мені стежити за найбільш важливими факторами успіху в проекті "Стартап"[11]. Перш за все, я провела невеликий уявний експеримент, в якому я, звичайно, робила нотатки. Я запитала себе, як засновника, чи могла б я пояснити незнайомцеві суть моєї бізнес-ідеї протягом трьох-п'яти хвилин? У цьому описі я повинна розглянути, як компанія повинна

виділятися серед конкурентів при запуску стартапу. Я повинна бути гранично ясна, коли розповідаю про переваги для моїх майбутніх клієнтів і надихати усіх, хто мене слухає.

Від магазину одягу, до смарт офісу та системи контролю бізнесу. Звичайно, я також є одним із засновників, який дуже захоплений відкриттям бізнесу і стурбований успіхами мого стартапу і перевагами для клієнтів. Дуже важливий регулювальний гвинт, щоб буквально "перевернути" успіх компанії – це маркетинг та реклама. Багато стартапів роблять помилку, недооцінюючи маркетинг в самому початку. Тут ви повинні підібрати цільові групи, хто вони та де вони знаходяться. Звичайно, тип реклами, на яку орієнтований проект, цілком залежить від бізнесу який ви починаєте і як і де ви хочете вести бізнес. Наприклад, компанія, яка в основному живе в пішохідному русі, навряд чи зможе вижити без привабливого дизайну вітрин. Однак консалтингова фірма, яка шукає поради по складним і конфіденційним питань, може багато чого зробити, наприклад, з широким спектром інформації в Інтернеті. У більшості випадків наявність потужного веб-сайту є однією з найбільш важливих маркетингових заходів при запуску стартапу. Найкраще покладатися на рішення, яке реалізовано професійно. Гарний і сучасний дизайн і функціональна система управління контентом є важливими критеріями вибору. Експерти в області маркетингу можуть допомогти розробити «продаваючий» сайт. Навіть якщо вони стоять багато грошей, ви повинні подумати про ці інвестиції. Добре підтримувана та інформативна присутність в Інтернеті дозволить моїй компанії виглядати серйозно відразу після її заснування, та давати надію на чесність компанії користувачам, які вперше зайшли на сайт для перегляду інформації про продукт.

Починаючи вести свій стартап проект, потрібно мати багато навичок і бути готовим до навчання. Якщо ви відчуваєте, що готові до цього, ви можете зробити що-небудь, щоб полегшити свій старт. Ймовірно, найбільш важливим документом, який ви можете побудувати для цієї мети, є бізнес-план. Деякі зі згаданих вище моментів, такі як бізнес-ідея і маркетинговий план, допоможуть вам у наступних питаннях:

- Хто належить до команди засновників проекту?
- Як ви оцінюєте ринок і конкуренцію після старту роботи проекту?
- Яке початкове місце розташування ви вибрали для запуску?
- Які ризики, сильні та слабкі сторони має ваша бізнес-ідея і ваш майбутній продукт?

До цього додається фінансовий план, який максимально точно розрахований до старту. При запуску стартапу майже завжди виникає необхідність у фінансуванні. Звичайно, тут є велика різниця: деякі компанії мають потребу в дорогих машинах або обладнанні, в той час як інші можуть працювати лише з ноутбуком, програмним забезпеченням і творчим підходом. Якщо ви не можете або не хочете запускати свої власні ресурси, ви можете використовувати різні джерела фінансування при налаштуванні стартапу. Особливо в Києві, стартапи знаходять різні варіанти підтримки, тому що там, зазвичай проводиться багато виставок та ярмарок стартапів.

Стартап став популярним, завдяки своїй гнучкості та набув популярності у Європі та світі через значне зниження фінансових стін для виходу на ринок. Також, завдяки інтернету, клієнтів стало знаходити набагато легше та швидше. Комунікація з ними також вийшла на новий рівень завдяки Basecamp, Slack, Trello, Asana і т.д.

7.1 Опис ідеї стартап проекту

В перших трьох пунктах я описала уявлення про мій проект, зміст моєї ідеї, ризики, ринки та зобразила це у таблиці (Таблиця 7.1).

Таблиця 7.1 – Зміст ідеї стартап-проекту

Зміст ідеї	Напрямки застосування	Вигоди для користувача
------------	-----------------------	------------------------

<p>Розробити незалежну систему. Запропонувати нове готове рішення “під ключ” для управління малим та середнім бізнесом; включаючи управління складським обліком і його логістикою, зберігання усіх даних в хмарному сховищі та додатки для замовлення, перегляду товарів з складу для користувачів Apple iOS та Android приладів будь-якої діагоналі.</p> <p>Кваліфікована подальша підтримка системи спеціалістом.</p>	<p>1. Зберігання товарів будь-якого виду.</p>	<p>Завдяки розподілення поставок на Склад по “Видам”, є можливість зберігання товарів з швидким стікання строку давності.</p>
	<p>2. Віддалений моніторинг.</p>	<p>Завдяки API серверам, є можливість віддалено моніторити пересування у системі. Контролювати передачу товарів на інші склади, контролювати наявність товарів, актуальність інформації.</p>
	<p>3. Інтеграція системи в існуюче підприємство та додавання сторонніх модулів.</p>	<p>Архітектура передбачає інтеграцію в новий склад, та додавання нових інструментів завдяки модульній структурі.</p>

Завдяки проведеному аналізу конкурентного ринку, пропозицій конкурентів, існуючих аналогів моя система передбачає: визначення сильних та слабких сторін запропонованого мною проекту, визначення характеристик ідей проекту, перелік техніко-економічних характеристик ідей (Таблиця 7.2).

Таблиця 7.2 – Визначення характеристик ідеї-проекту

№ п/п	Техніко – економічні характеристики	(потенційні) товари/концепції конкурентів	W (слабка сторон	N (нейтра льно	S (сильн а
----------	---	---	------------------------	----------------------	------------------

	ідеї	Мій проект	Конкурентний проект	а)	сторона а)	сторона
1	Пропозиція продажу ПЗ або оренда ПЗ для автоматизації управління складським обліком	Дає змогу	Дає змогу			V
2	Підвищення швидкості обробки замовлення на складі за допомогою додатків	Дає змогу	Не дає змогу			V
3	Гнучке налаштування	Дає змогу	Частково дає змогу			V
4	Розповсюдженість програмного забезпечення	Дає змогу	Дає змогу	V		
5	Підтримка будь якої операційної системи	Дає змогу	Не дає змогу		V	

7.2 Технологічний аудит проекту

Аналіз складових проекту передбачає визначення технологічної здійсненності ідеї проекту. Аналіз складових прописано в Таблиці 7.3.

Таблиця 7.3 – Технологічна здійсненність ідеї проекту

№ п/п	Ідея проекту	Технології її реалізації	Наявність технології	Доступність технології
1	Розробити систему з зрозумілим інтерфейсом для автоматизації підприємства	Мова програмування C#	Наявна	Доступна
2	Розробити програмний додаток для користувачів платформ Android	Мова програмування Java	Частково наявна	При обмеженому бюджеті недоступна
3	Розробити програмний додаток для користувачів платформ iOS	Мова програмування Swift	Наявна	Доступна

7.3 Аналіз ринкових можливостей запуску

Під час впровадження будь-якого проекту, на ринку існують низка загроз, перешкод для коректної реалізації проекту. Визначення ринкових можливостей дозволить нам урахувати стан ринкового середовища, потреби потенційних клієнтів та урахувати конкурентів. Для початку проводиться аналіз попиту проекту.

У Таблиці 7.4 наведено попередню характеристику потенційного ринку стартап-проекту.

№ п/п	Показник стану ринку (найменування)	Характеристика

1	Кількість головних гравців, од	10
2	Загальний обсяг продаж, грн/ум.од	20 тис. ум. од. в місяць
3	Динаміка ринку (якісна оцінка)	Зростає
4	Наявність обмежень для входу(вказати характер обмежень)	Зацікавленість потенційних клієнтів, початковий капітал до 10 тис.
5	Специфічні вимоги до стандартизації та сертифікації	Немає
6	Середня норма рентабельності в галузі (або по ринку), %	50% в рік

Далі я визначила потенційні групи клієнтів для продукту, визначила його характеристики та надала їх у таблиці 7.5 , де показано характеристику потенційних клієнтів стартап-проекту.

Таблиця 7.5 Характеристика потенційних клієнтів стартап-проекту

№ п/п	Потреба, що формує ринок	Цільова аудиторія (цільові сегменти ринку)	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів до товару

1	Відсутність системи управління обліком к віддаленим керуванням з смартфону	Малий та середній бізнес	Кожна з потенційних цільових груп клієнтів має свої вимоги до стандартів програмного забезпечення такого типу	Відповідність результату найвищим стандартам якості на ринку
2	Зменшення затрат часу на вистежування товару	Малий та середній бізнес	Використання системи на етапі розробки програмного забезпечення	Зменшення затрат часу на вистежування товару на підприємстві
3	Зменшення затрат на впровадження нових модулів	Малий та середній бізнес	Використання системи на етапі розробки програмного забезпечення	Зменшення затрат на впровадження нових модулів

Після того, як ми визначили потенційні групи клієнтів, проводимо аналіз ринку. У Таблиці 7.6 показані фактори загроз реалізації стартап-проекту.

Таблиця 7.6 Фактори загроз

№ п/п	Фактор	Зміст загрози	Можлива реакція компанії
1	Незацікавленість клієнтів	Внаслідок невдалої реклами та	Внесення додаткових рекламних

		маркетингу, потенційний клієнт може не зацікавитись продуктом	продуктів, сервісних послуг до маркетингового плану
2	Програш в конкуренції	Втрата звання надійності	Удосконалення продукту, додання нових унікальних інструментів, грамотна цінова політика.

У Таблиці 7.7 показано фактори можливостей при реалізації стартап-проекту.

Таблиця 7.7 Фактори можливостей

№ п/п	Фактор	Зміст можливості	Можлива реакція компанії
1	Перехід до домінування на ринку WMS систем	Зростання попиту на наші послуги	Удосконалення системи, додання нових унікальних можливостей, якісне та кількісне нарощування потужностей.

У наступній Таблиці 7.8 зображено особливості конкурентного середовища, загальні ризики конкуренції на ринку та його вплив на впровадження нашого проекту до підприємства.

Таблиця 7.8 Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)
1. Конкуренція	Використання вже існуючих на ринку технологій.	Розробка кращого програмного продукту на ринку.
2. Локальний	Існує багато пропозицій, але немає єдиного національного постачальника послуг.	Окремий підхід до кожної локальної ділянки, локалізація документації.
3. Міжгалузєва	Програмний продукт фокусується на управлінні складськими приміщеннями підприємств.	Фокусування на конкретному напрямку розвитку.
4. Товарно-видова	Конкуренція з іншими пропозиціями на ринку.	Аналізування конкурентних програмних рішень.
5. Цінова	Виявлення причин упадку проекту.	Оновлення, додавання унікальних можливостей, покращення.

6. Марочна	Унікалізація назви продукту.	Контролювати документацію про авторське право на продукт.
------------	---------------------------------	---

У наступній Таблиці 7.9 зображено більш конкретний аналіз умов Конкуренції в галузі.

Таблиця 5.9 Аналіз умов в галузі за М. Портером

Складові аналізу	Прямі конкуренти а галузі: компанії по розробці програмного забезпечення	Потенційні конкуренти	Постачальники	Клієнти	Товари замінники
	Перелік прямих конкурентів: SAP, we clapp, Actindo	Бар'єри входження на ринок програмного забезпечення: гнучкі ціна	Залучення малопопулярних постачальників, ефективні канали збуту в інтернеті	Чутливість до змін цін на ринку	Ціна продукту, пропозицій з боку ПЗ та функціонал

Висновки:	Середня інтенсивність	Можливість виходу на ринок є	Постачальники не диктують цінову політику на ринку	Клієнти диктують вимоги до якості продукту, до ціни, зручності та функціональності	Товари змінники не обмежують роботу
-----------	-----------------------	------------------------------	--	--	-------------------------------------

На основі наведеного вище аналізу конкуренції на ринку, визначимо та обґрунтуємо перелік факторів конкурентоспроможності (Таблиця 7.10).

Таблиця 7.10 Обґрунтування факторів конкурентоспроможності

№ п/п	Фактор конкурентоспроможності	Обґрунтування (наведення чинників, що роблять фактор для порівняння конкурентних проектів значущим)
1	Автоматизація складського обліку	Можливість відстежування переміщень на складі віддалено, за допомогою додатку
2	Надання сервісних послуг	Сервісна підтримка програмної частини

У Таблиці 7.11 наведено сильні та слабкі сторони розробленого проекту.

Таблиця 7.11 – Порівняльний аналіз сильних та слабких сторін продукту

№ п/п	Фактор конкурентоспроможності	Бали 1-20	Рейтинг товарів-конкурентів у порівнянні з розробленим програмним продуктом							
			-3	-2	-1	0	+1	+2	+3	
1	Раціональний ціновий показник	15	+							
2	Надання сервісних послуг	20			+					
3	Періодична діагностика	7					+			
4	Необхідність залучення висококваліфікованих кадрів	10						+		

У Таблиці 7.12 наведено фінальний етап ринкового аналізу можливостей впровадження проекту. Наведено SWOT-аналіз.

Таблиця 7.12 – SWOT – аналіз стартап-проекту.

<p>Сильні сторони:</p> <ul style="list-style-type: none"> – Автоматизування управління підприємством – Раціональний ціновий показник – Віддалене керування 	<p>Слабкі сторони:</p> <p>Необхідність залучення кваліфікованих кадрів, популярність рішення</p>
<p>Можливості:</p> <p>Ріст попиту на ексклюзивні можливості продукту.</p>	<p>Загрози:</p> <p>Незацікавленість клієнтів, збільшення кількості конкурентів, зменшення популярності.</p>

Альтернативи ринкового впровадження системи наведені нижче(таблиця 7.13).

Таблиця 7.13 – Альтернативи ринкового впровадження стартап-проекту

№ п/п	Альтернатива (орієнтовний комплекс заходів) ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації
1.	Зменшення ціни програмного продукту.	менша	незначні
2.	Збільшення ціни програмного продукту у відповідності до розширення функціоналу системи.	більша	незначні
3.	Укладення договорів з Українськими компаніями-новачками та швидке захоплення ринку.	висока	незначні

Обрана альтернатива – збільшення ціни програмного продукту у відповідності з розширенням функціоналу стартап проекту.

7.4 Розроблення ринкової стратегії проекту

Обґрунтування вибору цільових груп потенційних споживачів наведено у Таблиці 7.14.

Таблиця 7.14 – Вибір цільових груп потенційних споживачів

№ п/п	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи (сегменту)	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
1.	Розробники програмного забезпечення	Високий	Середній	Середній	Високий
2.	Склади без автоматизованого обліку	Високий	Великий	Середній	Високий

В якості цільових груп було обрано: розробники програмного забезпечення та склади без автоматизованого обліку.

Визначення базової стратегії розписано у Таблиці 7.15.

Таблиця 7.15 – Визначення базової стратегії розвитку

№ п/п	Обрана альтернатива розвитку проекту	Стратегія охоплення ринку	Ключові конкурентоспроможні позиції відповідно до обраної альтернативи	Базова стратегія розвитку
1.	Збільшення ціни програмного продукту у відповідності до розширення функціоналу системи.	Охоплення усього ринку.	Автоматизованість системи, виявлення відхилень продуктивності.	Стратегія спеціалізації.
2	Дешевизна проекту	Раціональні витрати на послуги	Застосування загально вживаних рішень	Стратегія лідерства

Далі ми визначаємо базову стратегію конкурентної поведінки яка наведена у Таблиці 7.16.

Таблиця 7.16 – Визначення базової стратегії конкурентної поведінки

№ п/п	Чи є проект «першопрохідцем» на ринку?	Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?	Чи буде компанія копіювати основні характеристики товару конкурента, і які?	Стратегія конкурентної поведінки
1.	Ні	Забирати існуючих та шукати нових.	Буде частково копіювати основні характеристики конкурентів.	Стратегія заняття конкурентної ніші.

На основі вищесказаного, розробляється стратегія позиціонування (Таблиця 7.17)

Таблиця 7.17 – Визначення стратегії позиціонування

№ п/п	Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкуренто-спроможні позиції власного стартап-проекту	Вибір асоціацій, які мають сформувати комплексну позицію власного проекту (три ключових)
1	Легкість використання програмного продукту.	Конкурентна	Автоматизованість виявлення відхилень продуктивності.	Швидкий, гнучкий, зрозумілий.
2	Висока якість послуг	Стратегія диференціації	Новизна, гарантування якості продукту, підтримка.	Якість, надійність, точність.
3	Мінімальні витрати	Стратегія лідерства по витратах	Універсальність.	Універсальність, дешевизна.

7.5 Розроблення маркетингової програми

Ключові переваги концепції потенційного товару наведено нижче у Таблиці 7.18. Для цього, ми підсумували результати усіх попередніх досліджень та аналізів.

Таблиця 7.18 – Визначення ключових переваг концепції потенційного товару

№ п/п	Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами

1	Автоматизованість виявлення відхилень продуктивності.	Розробники мають інструмент для відстежування продуктивності додатка.	Виявлення упадку продукту.
2	Якість	Висока надійність даних та якість	Надійність
3	Дешевизна	Раціональне використання наданих клієнтом коштів	Дешевизна

Надалі у Таблиці 7.19 визначено три рівні маркетингової моделі товару.

Таблиця 7.19 – Опис трьох рівнів моделі товару

Рівні товару	Сутність та складові
1. Товар за задумкою: Система управління складським обліком.	Опис базової потреби споживача, яку задовольняє товар. Стандартизовано якісний товар та послуги. Потреба в автоматизації процесу виявлення першопричин упадку продуктивності додатку.
2. Товар у реальному виконанні	Характеристики: – функціонал автоматизації складського обліку на підприємстві; – функціонал віддаленого клерування процесами на підприємстві.
4. Товар із підкріпленням	До продажу: Програмні додатки. Після продажу: сервісна підтримка.

Визначення меж встановлення ціни на послугу описано у Таблиці 7.20

Таблиця 7.20 – Визначення меж встановлення ціни

№ п/п	Рівень цін на товари-замінники	Рівень цін на товари-аналоги	Рівень доходів цільової групи споживачів	Верхня та нижня межі встановлення ціни на товар/послугу
1	1000 у.о./од. (стандарт)	-	Високий	Н. 5000 у.о. (Товар) В. 11000 у.о. (Товар)

Формування оптимальної системи збуту, в межах якого приймається рішення описано у Таблиці 7.21.

Таблиця 7.21 – Формування системи збуту

№ п/п	Специфіка закупівельної поведінки цільових клієнтів	Функції збуту, які має виконувати постачальник товару	Глибина каналу збуту	Оптимальна система збуту
1	Купівля ліцензії на використання програмного продукту	Цілодобова доступність продукту на ресурсах	Без посередників	Договірна система збуту

На останок ми повинні розробити концепцію маркетингових комунікацій. Вона наведена у Таблиці 7.22.

Таблиця 7.22 – Концепція маркетингових комунікацій

№ п / п	Специфіка поведінки цільових клієнтів	Канали комунікацій, якими користуються цільові клієнти	Ключові позиції, обрані для позиціонування	Завдання рекламного повідомлення	Концепція рекламного звернення
1	Пошук потрібних програмних рішень в інтернеті	Мережні ресурси	Віддалене керування системою, точність та конфіденційність даних	Зацікавити в покращеному продукті	Демонстрація переваг
2	Зацікавленість потенціальних клієнтів в якісному та точному продукті з раціональним використанням ресурсів	Мережні ресурси	Гарантування якості, конфіденційності, стандартизація	Зацікавити у перспективі засилаючись на зростання попиту на данні програмні продуукти.	Демонстрація переваг додатків

3	Зацікавленість у великій кількості якісного, фінішного продукту	Мережні ресурси	Гарант якості програмного додатку, канали постачальників	Зацікавити в плюсах проекту та у глибині каналів постачання	Демонстрація переваг додатків
---	---	-----------------	--	---	-------------------------------

7.6 Висновки до розділу

Існує багато ризиків при запуску і в перші роки стартапу. Перш за все, це недосвідченість засновників. Оскільки вони часто дуже молоді, у них мало досвіду в корпоративному управлінні. Крім того, в перші дні часто неясно, чи можна розробити продукт необхідної якості. Також не можна передбачити, чи прийме ринок продукт взагалі.

Ці причини в основному відповідальні за той факт, що тільки дуже мало хто стартапи є успішними. Як правило, сім-вісім з десяти молодих компаній зазнають невдачі. Для решти 1 - 2 очікуване зростання відсутнє. Тільки 10 відсотків молодих підприємців зазвичай досягають бажаного успіху. Але, завдяки правильному проведеному аналізу ринку, можна зробити деякі висновки.

Комерціалізація стартап проекту для розвитку та впровадження запропонованого мною рішення, має великі шанси бути прибутковою системою, розростатися у майбутньому, на успішно створювати конкуренцію на ринку. Впровадження є перспективним, адже спочатку будуть охоплюватися малі та середні бізнеси, а згодом захоплювати ринок гігантів України. Конкретність обумовлена нижчою ніж є на даний момент, ціною на ринку, та більшим спектром послуг. Також додаток розрахований на розвиток, і вже зараз підтримує крос платформенність. Це вигідно вирізняє програмну пропозицію серед інших. Імплементация проекту доцільна, так як рентабельність та зацікавленість потенційних груп майбутніх клієнтів робить умови для розвитку стартап проекту – сприятливими.

ВИСНОВК

Під виконання магістерської дисертації я проаналізувала свою переметну область яка замається автоматизацією підприємств. Також, я дослідила і інші рішення в моїй галузі та визначила вимогу до майбутньої системи.

В результаті виконаної розробки можна зробити наступні висновки:

- При розробці системи був пройдений повний цикл проектування програми від постановки завдання для розробника до впровадження системи управління;
- Розроблена система має такі переваги над іншими системами:
 1. Додавання товару способом сканування штрих коду;
 2. Статистика по підприємству;
 3. Можливість віддаленого керування за допомогою мобільного додатку;
 4. Можливість безстрокового зберігання інформації;
 5. Пошук по системі.

Отже, створена система відповідає заявленим вимогам, усі поставлені задачі було виконано, система реалізована в повному обсязі.

Результати було апробовано та впроваджено систему у підприємство ТОВ «Наш зелений світ». Акт про впровадження додано.

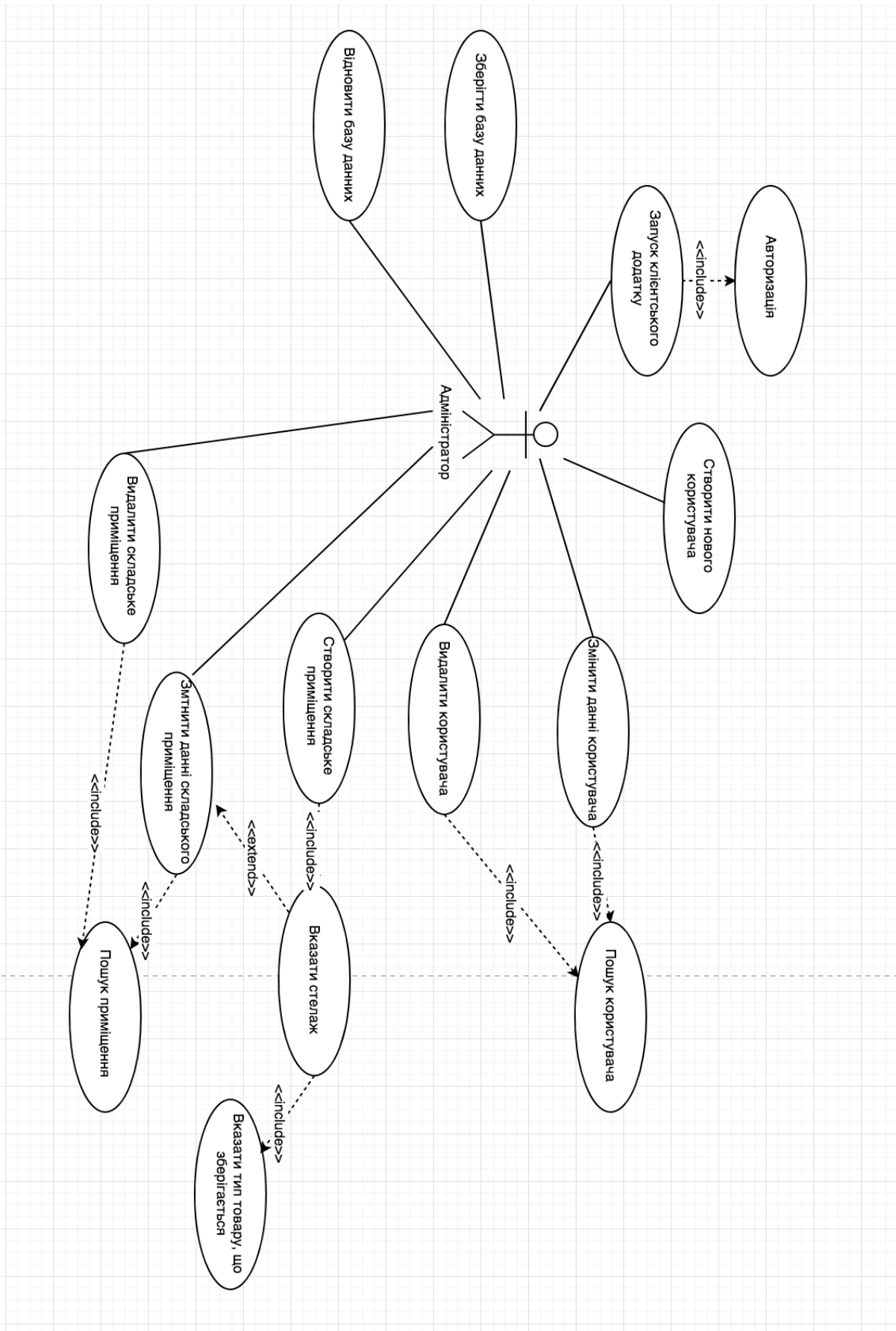
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Герберт Ш. С# 4.0. Полное руководство [Текст] / Ш. Герберт, — М. : Издательский дом «Вильямс», 2011. — 1056 с.
2. API Server [Электронный ресурс] – Режим доступа: <https://www.crossbase.de/DE/DE/loesungen/datenbereitstellung/api-server>
3. Усков А.А. Методические указания по выполнению курсового проекта по курсу «Разработка и стандартизация программных средств и информационных технологий». Смоленск: СФ АНО ВПО ЦС РФ "РУК", 2007.
4. Смирнова Г.Н. Проектирование экономических информационных систем: Учебник для студентов экономических вузов, обуч. по спец.: "Прикладная информатика в экономике", "Прикладная информатика в менеджменте", "Прикладная информатика в юриспруденции". - М.: Финансы и статистика, 2003. - 511 с.
5. Marc Unger 50 Dropship & Wholesale Vendors: Dropshipping List (Drop Shipping & Wholesalers Book 1) - 2015, 500с.
6. International Testing-Board Testautomatisierung Definition, Tutorial und Artikel Testautomatisierung [Электронный ресурс] / Code Project - Режим доступа до ресурсу: <https://www.testing-board.com/testautomatisierung/>
7. Маклаков С.В. Моделирование бизнес процессов с BPwin 4.0 М.: Диалог-МИФИ, 2002.
8. Realm Studio [Электронный ресурс] / Be moreproductive with Realm Studio – <https://realm.io/products/realm-studio/>
9. Соколов В.Ю. Інформаційні системи і технології: Навчальний посібник – Київ ДУІКТ 2010.

10. Amy Brown and Greg Wilson The Architecture of Open Source Applications – 2016. – 121с.
11. Start up [Электронный ресурс] / MySQL 2017 – Режим доступа до ресурсу: <https://www.startupticker.ch/en/news/june-2015/startup-projekte-haben-immer-dieselben-schwachen>
12. Модульне тестування (Unit testing) [Электронный ресурс] / Qalight 2016 – Режим доступа до ресурсу: <http://lviv.qalight.com.ua/baza-znan/modulne-testuvannya/>
13. SAP [Электронный ресурс] / Developer Overview – Режим доступа до ресурсу : <https://www.sap.com/index.html>
14. Gwynne Richards Warehous management: A Complete Guide to Improving Efficiency and Minimizing Costs in the Modern Warehouse – 2017 - 250с.

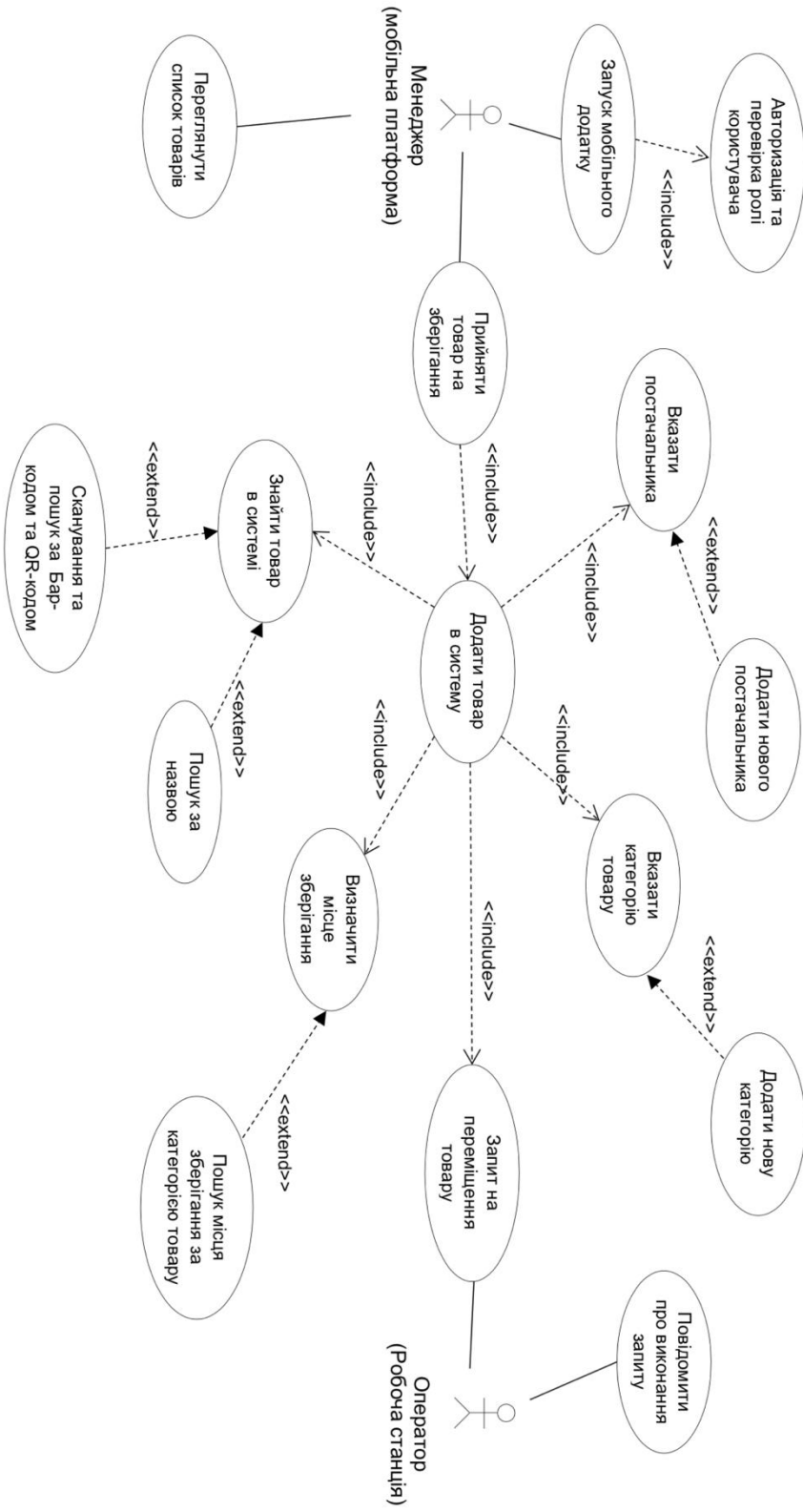
ДОДАТОК А

Use Case діаграма варіантів використання системи.

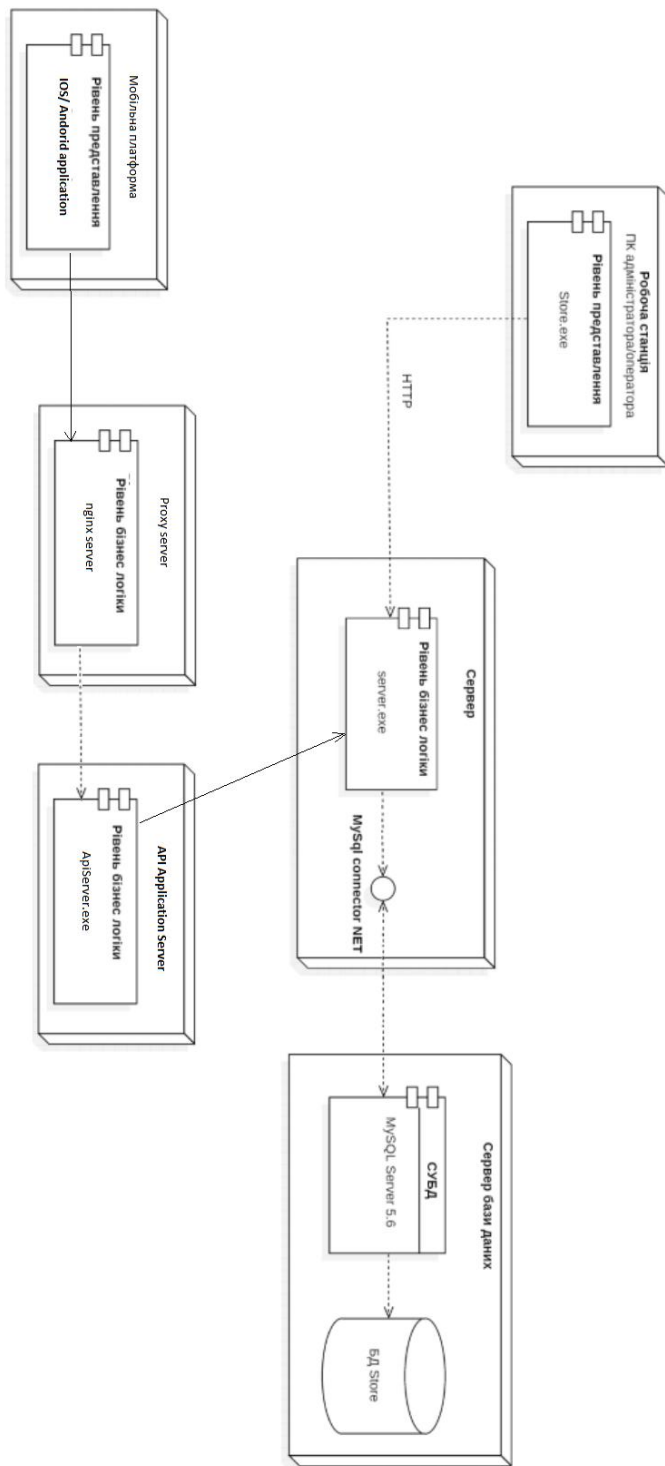


ДОДАТОК Б

USE Case діаграма варіантів використання системи мобільного додатку

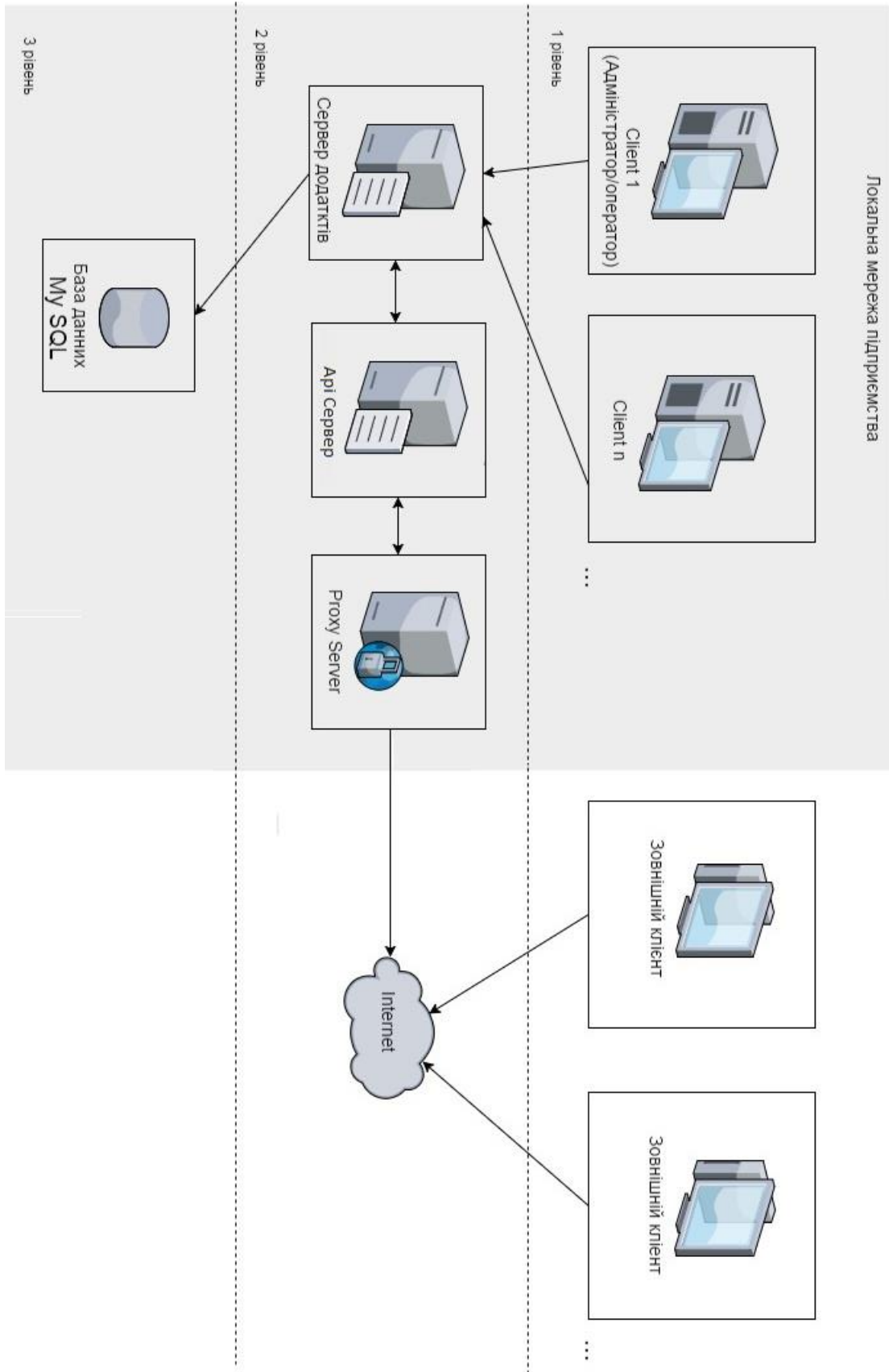


ДОДАТОК В
UML Діаграма розгортання



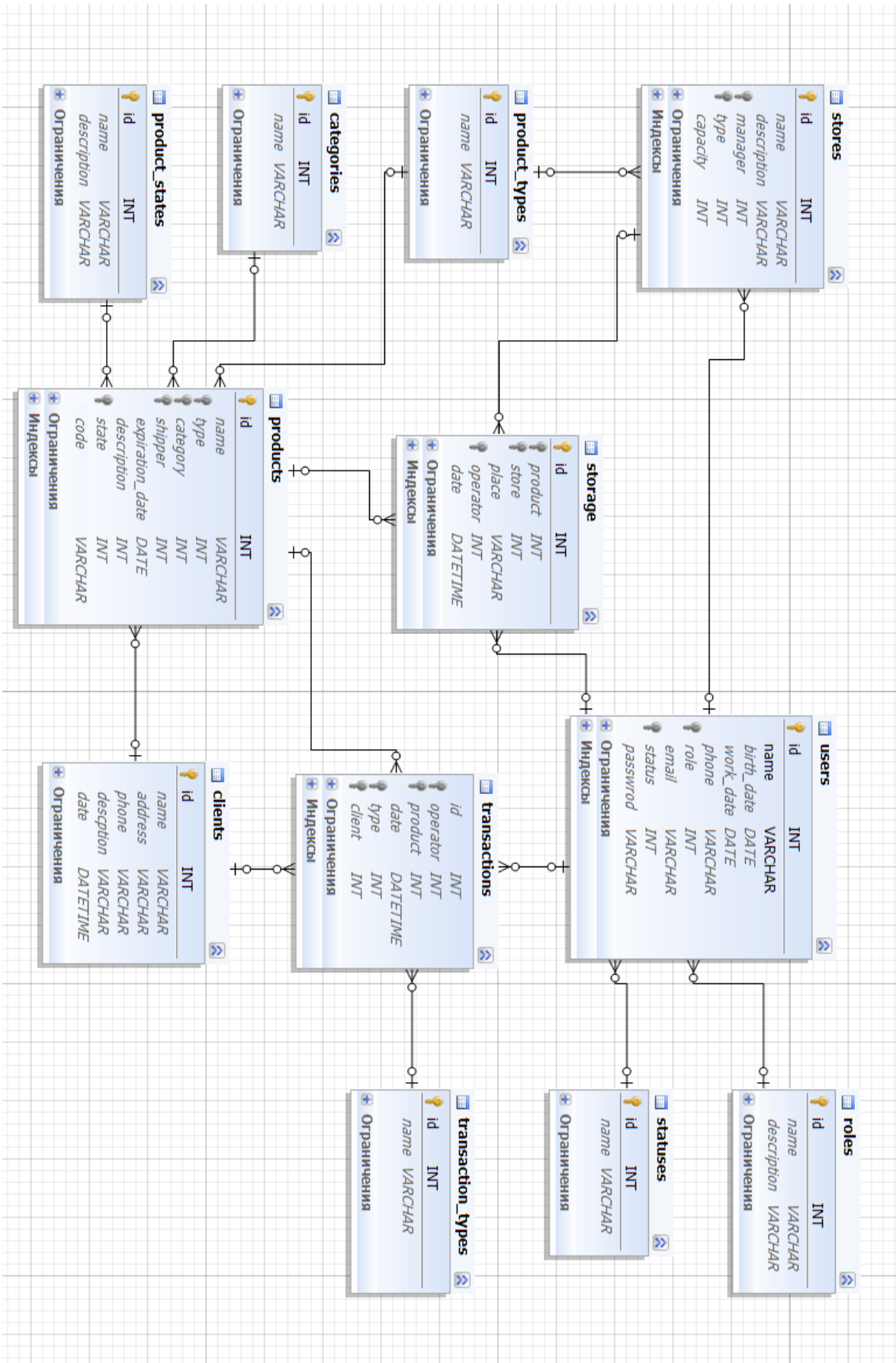
ДОДАТОК Г

Структурна схема системи



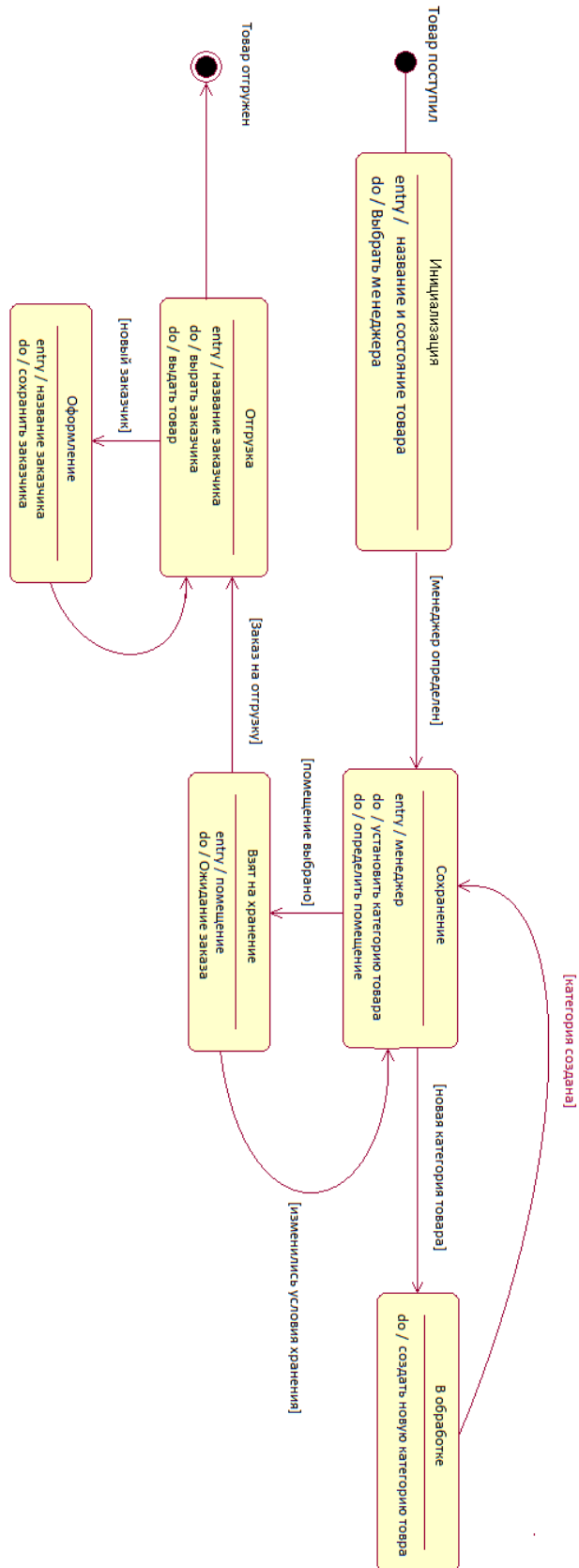
ДОДАТОК Д

ER діаграма бази даних



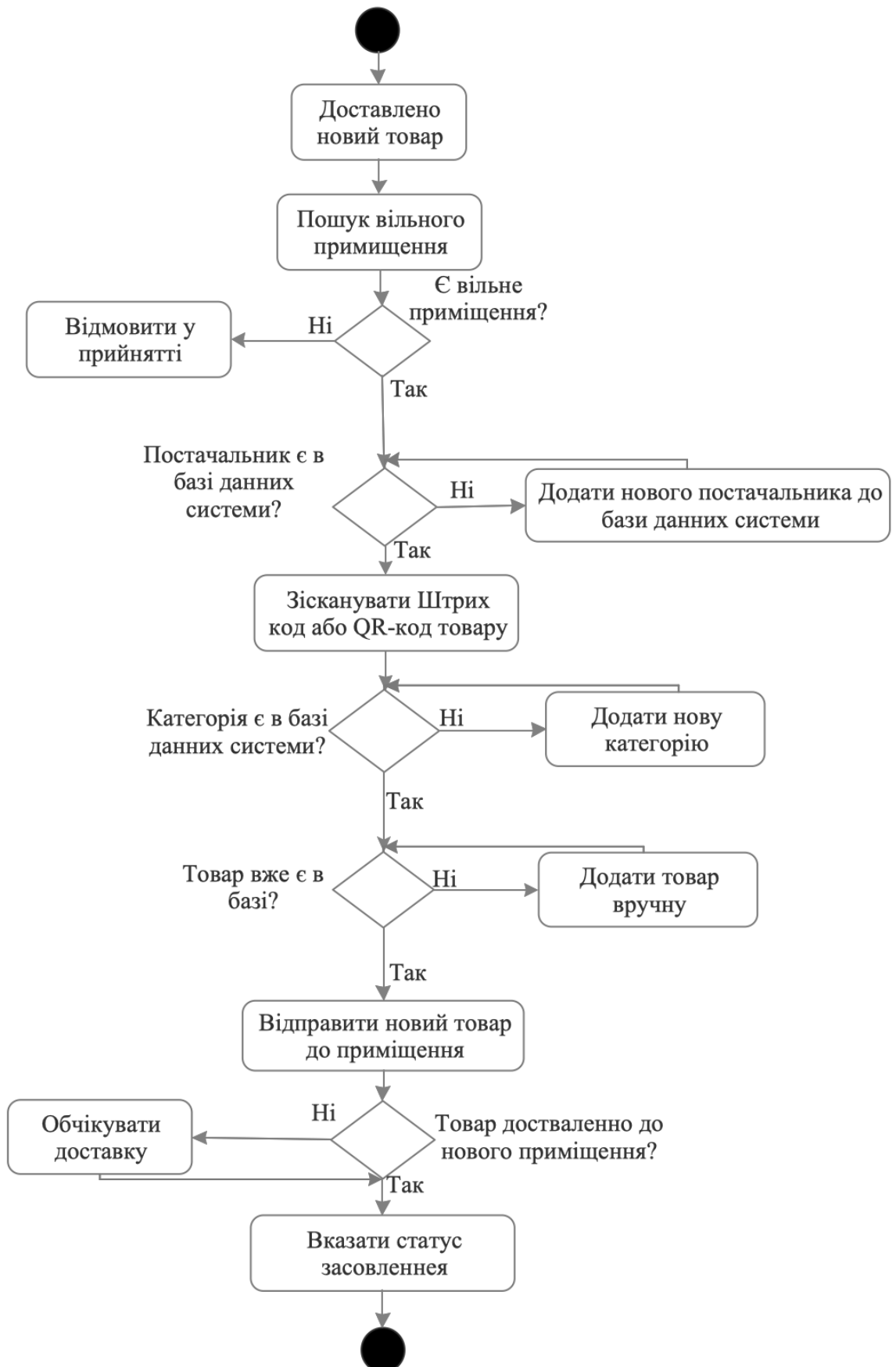
ДОДАТОК Е

UML діаграма станів



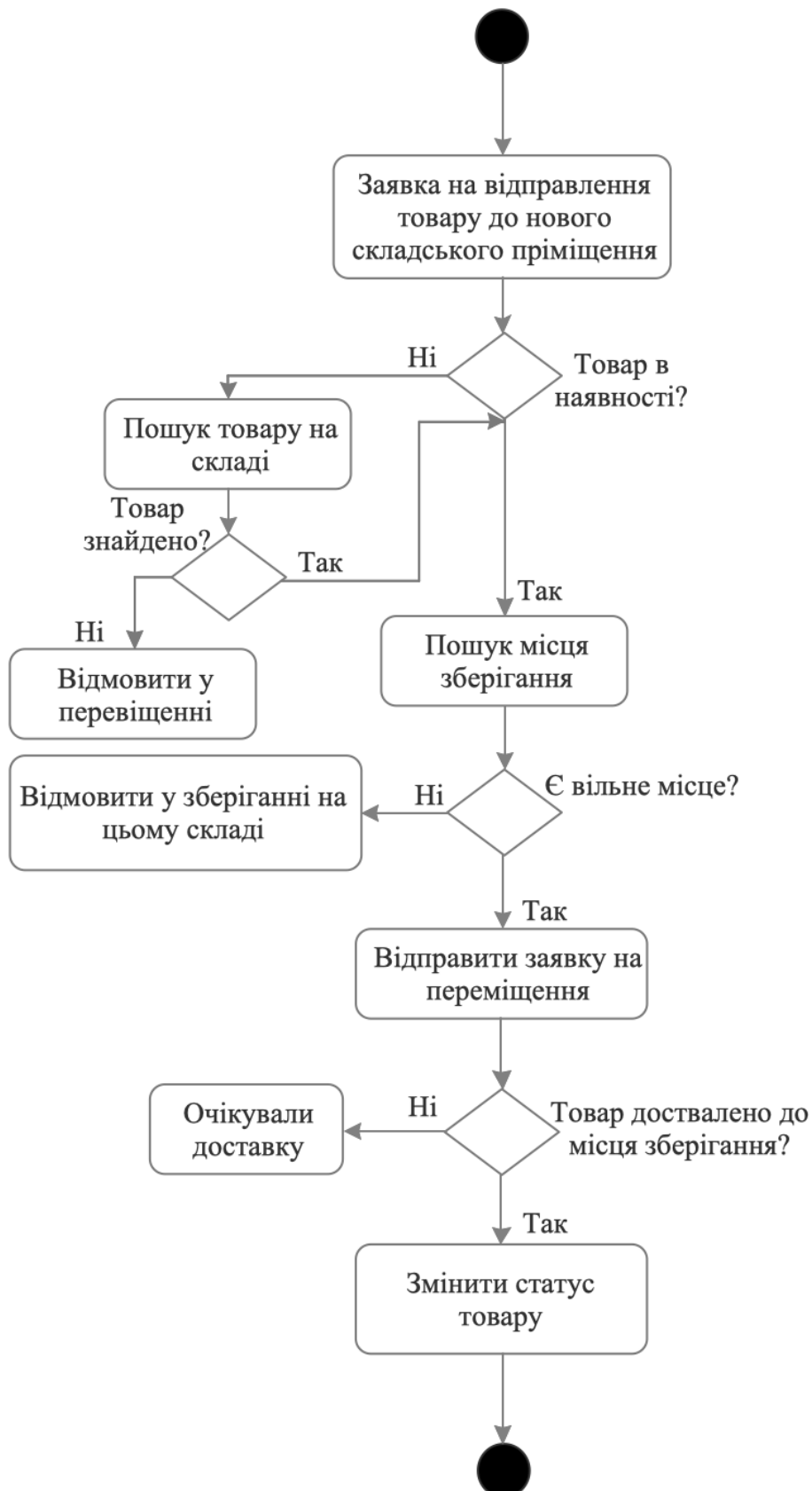
ДОДАТОК Ж

UML Діаграма активності. Процес доставки товару



ДОДАТОК Е

UML Діаграма активності. Процес переміщення товару по складу



ДОДАТОК К
Акт про впровадження