

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

**НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ КОРАБЛЕБУДУВАННЯ
імені адмірала Макарова**

Херсонський навчально-науковий інститут

Кафедра інформаційних технологій та фізико-математичних дисциплін

 “Допущений до захисту”
Завідувач кафедри
д.т.н., доцент Гучек П.Й.

“ ____ ” _____ 2025 р.

ПОЯСНЮВАЛЬНА ЗАПИСКА

до кваліфікаційної роботи

на здобуття ступеня вищої освіти “магістр”

на тему: “Дослідження методів та технологій управління бізнес-процесами
обслуговування клієнтів у сфері мобільного зв’язку та розробка
інформаційної системи”

Виконав: студентка VI курсу, групи 6157зм
спеціальності

122 - “Комп’ютерні науки”, ОПП -

(шифр і назва спеціальності та освітньо-професійної програми)

“Інформаційні управляючі системи та технології”

Савочкіна В.В.

(прізвище та ініціали)



Керівник

к.т.н., доц. Книрік Н.Р.

(прізвище та ініціали)



Рецензент

к.т.н., доц. Гайда А.Ю.

(прізвище та ініціали)



30.12 - 2025 року

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ КОРАБЛЕБУДУВАННЯ
імені адмірала Макарова
ХЕРСОНСЬКИЙ НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ**

Кафедра інформаційних технологій та фізико-математичних дисциплін

Освітній ступень магістр

Галузь знань 12 - "Інформаційні технології"
(шифр і назва)

Спеціальність 122 - "Комп'ютерні науки"
(шифр і назва)

Освітньо-професійна програма "Інформаційні управляючі системи та технології"

ЗАТВЕРДЖУЮ

Гарант освітньої програми



Гучек П.Й.

"23" жовтня 2025 р.

**ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ
на здобуття ступеня вищої освіти «магістр»**

Студенту

Савочкіної Вірі Володимирівні

(прізвище, ім'я, по батькові)

1. Тема кваліфікацій роботи: Дослідження методів та технологій управління бізнес-процесами обслуговування клієнтів у сфері мобільного зв'язку та розробка інформаційної системи.

Керівник роботи

Книрик Наталя Ромуальдівна, к.т.н., доцент

(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Тема затверджена розпорядженням по інституту від "17" жовтня 2025 року № 20

2. Термін подання студентом роботи 12.12.2025 р.

3. Вихідні дані до роботи:

4. Зміст та структура розрахунково-пояснювальної записки (перелік питань, які потрібно розробити):

4.1 Титульний аркуш, завдання на магістерську роботу, анотація (українською, російською, англійською), зміст, перелік умовних означень, символів, одиниць та термінів (при необхідності).

4.2 Вступ.

4.3 Дослідження технологій проектування інформаційних систем та аналіз їх особливостей.

4.4 Формування вимог до інформаційної системи, розробка її концепції і постановка задачі.

4.5 Розробка проектних рішень по інформаційній системі

4.6 Реалізація проекту інформаційної системи (технічна документація по проекту).

4.7 Розділ з економіки

4.8 Розділ з охорони праці та безпеки в надзвичайних ситуаціях (охорони оточуючого середовища).

4.9 Висновки.

4.10 Список використаної літератури.

4.11 Додатки (технічне завдання, загальний опис ІС, інструкція користувача, текст програми).

5. Перелік презентаційних матеріалів

5.1 Схема організаційної структури підприємства.

5.2 Архітектура автоматизованої ІС.

5.3 Організаційно-функціональна структура автоматизованої ІС.

5.4 Інформаційна модель програмного забезпечення автоматизованої ІС.

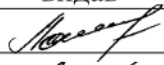
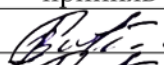
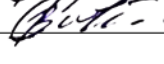
5.5 Модель переходів станів програмного забезпечення автоматизованої ІС.

5.6 Модель технологічного забезпечення автоматизованої ІС.

5.7 Інтерфейс програми

5.8

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
5.7	Ломоносов Дмитро Анатолійович		
5.8	Щедролосєв Олександр Вікторович		

7. Дата видачі завдання 27 жовтня 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Строк виконання етапів роботи	Примітка
1.	Вивчення і аналіз предметної галузі	28.10.2025	
2.	Розробка концепції ІС	30.10.2025	
3.	Розробка проектних рішень по ІС	31.10.2025	
4.	Реалізація проекту, розробка робочої документації	03.11.2025	
5.	Вирішення питань з економіки	06.11.2025	
6.	Вирішення питань охорони праці та безпеки в надзвичайних ситуаціях (охорони оточуючого середовища)	12.11.2025	
7.	Виконання графічної частини роботи	22.11.2025	
8.	Оформлення пояснювальної записки	06.12.2025	
9.	Подання роботи на попередній захист	12.12.2025	
10.	Подання роботи рецензенту	20.12.2025	
11.	Подання роботи на державний захист	29.12.2025	

Студент


Підпис

Савочкіна В.В

Прізвище та ініціали

Керівник роботи


Підпис

Книрик Н.Р.

Прізвище та ініціали

АНОТАЦІЯ

У роботі здійснено дослідження сучасних методів та технологій управління бізнес-процесами обслуговування клієнтів у сфері мобільного зв'язку. Розглянуто теоретичні засади моделювання та оптимізації бізнес-процесів, а також проаналізовано практичні інструменти автоматизації взаємодії з клієнтами. Особливу увагу приділено питанням підвищення ефективності роботи операторів мобільного зв'язку шляхом впровадження інформаційних систем, що забезпечують прозорість процесів, скорочення часу обслуговування та підвищення рівня задоволеності клієнтів.

Розроблено концепцію та реалізовано прототип інформаційної системи продажу мобільних телефонів, яка інтегрує функції управління замовленнями, обліку товарів, аналітики продажів та підтримки клієнтів. Система побудована з використанням сучасних технологій програмування та баз даних, що дозволяє забезпечити масштабованість, надійність та зручність у використанні.

Практична цінність роботи полягає у можливості застосування запропонованих рішень для вдосконалення бізнес-процесів компаній мобільного зв'язку та торговельних підприємств, що спеціалізуються на реалізації мобільних пристроїв. Результати дослідження можуть бути використані для підвищення конкурентоспроможності організацій, оптимізації витрат та покращення якості обслуговування клієнтів.

Ключові слова: управління бізнес-процесами, сфера мобільного зв'язку, оптимізація, автоматизація, інформаційна система.

ABSTRACT

This paper examines modern methods and technologies for managing customer service business processes in the mobile communications sector. The theoretical foundations of business process modelling and optimisation are considered, and practical tools for automating customer interactions are analysed. Particular attention is paid to improving the efficiency of mobile operators through the implementation of information systems that ensure process transparency, reduce service time and increase customer satisfaction.

A concept has been developed and a prototype of an information system for selling mobile phones has been implemented, which integrates the functions of order management, inventory accounting, sales analytics and customer support. The system is built using modern programming and database technologies, which ensures scalability, reliability and ease of use.

The practical value of the work lies in the possibility of applying the proposed solutions to improve the business processes of mobile communication companies and retail enterprises specialising in the sale of mobile devices. The results of the research can be used to increase the competitiveness of organisations, optimise costs and improve customer service quality.

Keywords: business process management, mobile communications, optimization, automation, information system.

ЗМІСТ

Вступ.....	7
РОЗДІЛ 1 ТЕОРЕТИЧНІ ТА ПРАКТИЧНІ ОСНОВИ УПРАВЛІННЯ БІЗНЕС-ПРОЦЕСАМИ В КОНТЕКСТІ ОБСЛУГОВУВАННЯ КЛІЄНТІВ ТЕЛЕКОМУНІКАЦІЙНИХ КОМПАНІЙ ТА РОЗДРІБНОЇ ТОРГІВЛІ.....	9
1.1. Актуальність теми та постановка проблеми дослідження	9
1.2. Особливості бізнес-процесів обслуговування клієнтів у телекомунікаційній галузі та ритейлі електроніки	11
1.3. Аналіз сучасних методологій управління бізнес-процесами (BPM) для гібридних операцій.....	12
1.4. Огляд ринку інформаційних систем та технологій для автоматизації продажів та обслуговування.....	13
1.5. Аналіз підприємства для якого виконується автоматизація	14
1.6. Визначення вимог до інформаційної системи продажу мобільних телефонів	17
1.7. Висновки до розділу 1	18
РОЗДІЛ 2 РОЗРОБКА ПРОЕКТНИХ РІШЕНЬ	20
2.1 Інформаційне забезпечення	20
2.2 Програмне забезпечення	28
Висновок до розділу 2	33
РОЗДІЛ 3 РЕАЛІЗАЦІЯ ПРОЕКТУ інформаційної СИСТЕМИ.....	34
Висновок до розділу 3	35
РОЗДІЛ 4. РОЗРАХУНОК ЕКОНОМІЧНОГО ЕФЕКТУ ВІД ВПРОВАДЖЕННЯ ІС ...	36
РОЗДІЛ 5 ОХОРОНА ПРАЦІ	43
5.1 Аналіз потенційно небезпечних і шкідливих факторів у приміщенні відділу кадрів.....	44
5.2 Розрахунок освітлення	50
Висновок до розділу 5	51
РОЗДІЛ 6. ОХОРОНА НАВКОЛИШНЬОГО СЕРЕДОВИЩА	52
Висновки	59
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	60
ДОДАТОК А – Технічне завдання	63
ДОДАТОК Б – Інструкція користувача	67
ДОДАТОК В – Технічний опис.....	68
ДОДАТОК Г – SQL скрипт для розробленої БД	70

ВСТУП

Сучасний етап розвитку економіки та інформаційних технологій характеризується високим рівнем конкуренції у сфері мобільного зв'язку та стрімким зростанням вимог клієнтів до якості обслуговування. Ефективне управління бізнес-процесами стає ключовим чинником успішності підприємств, що працюють у цій галузі. В умовах цифровізації та глобалізації ринку мобільні оператори та торговельні компанії змушені впроваджувати інноваційні технології, які забезпечують швидкість, прозорість та персоналізацію взаємодії з клієнтами.

Особливого значення набуває розробка та використання інформаційних систем, що дозволяють автоматизувати процеси продажу мобільних телефонів, оптимізувати управління замовленнями та підвищити рівень задоволеності клієнтів. Такі системи не лише сприяють зниженню витрат і підвищенню продуктивності, але й створюють конкурентні переваги на ринку.

Актуальність теми дослідження зумовлена необхідністю пошуку ефективних методів управління бізнес-процесами у сфері мобільного зв'язку та впровадження сучасних інформаційних технологій для підтримки процесів продажу.

Метою роботи є дослідження методів та технологій управління бізнес-процесами обслуговування клієнтів у сфері мобільного зв'язку та розробка інформаційної системи продажу мобільних телефонів.

Для досягнення поставленої мети визначено такі завдання:

- проаналізувати сучасні методи управління бізнес-процесами у сфері мобільного зв'язку;
- дослідити технології автоматизації процесів обслуговування клієнтів;
- розробити концепцію інформаційної системи продажу мобільних телефонів;
- реалізувати прототип системи та оцінити її ефективність.

Наукова новизна роботи полягає у комплексному підході до дослідження бізнес-процесів обслуговування клієнтів та інтеграції їх з інформаційною системою продажу мобільних пристроїв.

Практичне значення отриманих результатів полягає у можливості використання розробленої системи для підвищення ефективності діяльності підприємств мобільного зв'язку та торговельних компаній, що реалізують мобільні телефони.

РОЗДІЛ 1

ТЕОРЕТИЧНІ ТА ПРАКТИЧНІ ОСНОВИ УПРАВЛІННЯ БІЗНЕС-ПРОЦЕСАМИ В КОНТЕКСТІ ОБСЛУГОВУВАННЯ КЛІЄНТІВ ТЕЛЕКОМУНІКАЦІЙНИХ КОМПАНІЙ ТА РОЗДРІБНОЇ ТОРГІВЛІ

1.1. Актуальність теми та постановка проблеми дослідження

Сфера телекомунікацій, зокрема мобільного зв'язку, перебуває в стані постійної та динамічної трансформації. Конкуренція давно вийшла за межі якості сигналу та ціни на хвилини. Сьогодні ключовими драйверами росту є побудова екосистеми послуг навколо абонента, де мобільний зв'язок, інтернет, контент-сервіси та продаж обладнання (мобільних телефонів) утворюють єдиний цикл взаємодії. Ринок мобільних телефонів, особливо в сегменті продажу через канали телеком-операторів (у зв'язку з контрактами, розстрочками, бонусними програмами), є критично важливим джерелом прибутку та потужним інструментом утримання клієнтів.

Проте ця інтеграція створює комплексні виклики в управлінні **бізнес-процесами обслуговування клієнтів (БП ОК)**:

- **Фрагментація досвіду клієнта:** Абонент, який купує телефон у салоні оператора, стикається з розділеними процесами: консультація щодо телефону, оформлення кредиту/розстрочки, активація SIM-карти, налаштування послуг, реєстрація гарантії, подальше сервісне обслуговування. Часто ці процеси ведуться в різних, слабо інтегрованих системах.

- **Втрата інформації та неефективність:** Дані про клієнта, характеристики проданого телефону, деталі гарантійного зобов'язання, історія звернень можуть "губитися" між CRM, системами логістики та складу, фінансовими програмами та системами обслуговування мережі.

- **Необхідність комплексних рішень:** Сучасний клієнт очікує "одного вікна" — можливості в одному місці та в один етап вирішити всі питання: обрати тариф, підібрати оптимальний телефон з урахуванням бюджету, оформити фінансову угоду, активувати послуги та отримати технічну підтримку.

- **Відсутність єдиної платформи для продажу:** Процеси продажу телефонів часто автоматизовані недостатньо, що призводить до помилок у залишках на складі, проблемах із гарантійним обслуговуванням, ручному введенню даних та повільному обслуговуванню.

Наукова проблема полягає у відсутності цілісного методологічного підходу та адаптованих інформаційних рішень для інтеграції та автоматизації складних бізнес-процесів обслуговування клієнтів, що поєднують телекомунікаційні послуги та роздрібну торгівлю електронними пристроями.

Практична проблема — низька ефективність та клієнтоорієнтованість процесів продажу мобільних телефонів у салонах зв'язку через роздробленість систем, ручні операції та відсутність єдиного інформаційного простору.

Об'єкт дослідження: бізнес-процеси обслуговування клієнтів у сферах мобільного зв'язку та роздрібної торгівлі мобільними телефонами.

Предмет дослідження: методи, технології та інформаційні системи для управління та автоматизації інтегрованих БП ОК.

Мета роботи: дослідити сучасні методи та технології управління БП ОК у контексті інтеграції телеком-послуг і роздрібних продажів, та на цій основі розробити проект інформаційної системи для автоматизації процесів продажу мобільних телефонів у салонах зв'язку.

1.2. Особливості бізнес-процесів обслуговування клієнтів у телекомунікаційній галузі та ритейлі електроніки

Бізнес-процеси в даному контексті мають гібридну природу, поєднуючи риси обох галузей:

1. **Високий рівень регуляторних вимог (телеком-частина):** Обов'язкова ідентифікація абонента, оформлення договорів згідно із законодавством про телекомунікації, облік тарифікації.

2. **Логістика та управління товарними запасами (ритейл-частина):** Необхідність точного контролю залишків телефонів, їх варіантів (модель, колір, пам'ять), серійних номерів, управління поставками та віртуальними складами в салонах.

3. **Фінансові сервіси (інтеграційна складова):** Процеси оформлення розстрочки, кредитування, страхування телефонів, робота з партнерами-банками, що потребує безпечного обміну даними.

4. **Складність циклу життя продукту:** Проданий телефон має подвійний життєвий цикл: як товар (гарантія, ремонт, trade-in) і як частина телеком-сервісу (прив'язка до мережі, налаштування інтернет-послуг).

Ключові БП ОК, що підлягають дослідженню та автоматизації:

- **Процес комплексного продажу "Телефон + Тариф":** Від консультації до видачі товару та активації послуг.
- **Процес фінансового супроводу:** Оформлення розстрочки/кредиту, перевірка клієнта через кредитні бюро, підписання електронних документів.
- **Процес гарантійного та постгарантійного обслуговування:** Реєстрація гарантії, прийом телефону в ремонт, логістика до сервісного центру, комунікація з клієнтом.

- **Процес Trade-in (прийом старого телефону в заміну):** Оцінка старого пристрою, перерахування його вартості, утилізація даних.
- **Процес управління лояльністю:** Нарахування бонусів за покупку телефону, інтеграція з програми лояльності оператора.

1.3. Аналіз сучасних методологій управління бізнес-процесами (BPM) для гібридних операцій

Для управління такими складними процесами застосовні:

- **BPMN 2.0 (Business Process Model and Notation):** Стандарт де-факто для візуального моделювання процесів. Дозволяє чітко відобразити паралельні гілки (наприклад, одночасне оформлення кредиту та резервування телефону на складі), точки прийняття рішень, взаємодію з різними системами (за допомогою позначень для сервісних задач - Service Tasks).
- **Методологія BPM (Business Process Management):** Її цикл (Design, Model, Execute, Monitor, Optimize) ідеально підходить для постійного вдосконалення кросканальних процесів на основі даних про їхню ефективність (KPI: середній час продажу, конверсія, % повернень).
- **Принципи Customer Journey Mapping (CJM):** Доповнюють BPM, дозволяючи проектувати процеси не з точки зору компанії, а з точки зору переживань клієнта на кожному етапі (вибір, покупка, використання, підтримка).
- **Архітектурний фреймворк eTOM (enhanced Telecom Operations Map):** Його потоки в областях "Product & Offer Management" та "Customer Relationship Management" можуть бути розширені для включення етапів управління товарними запасами та продажами обладнання.

Оптимальним підходом є комбінація BPMN для моделювання логіки процесів, філософії BPM для їх життєвого циклу та принципів CJM для забезпечення клієнтоцентричності.

1.4. Огляд ринку інформаційних систем та технологій для автоматизації продажів та обслуговування

Аналіз показав наявність декількох категорій рішень, кожне з яких має обмеження:

1. **Традиційні телекомунікаційні BSS/CRM (Amdocs, Salesforce Telecom Cloud):** Мають потужні можливості для управління абонентами та тарифами, але їх модулі для роздрібної торгівлі товарами часто є "надбудовами" з обмеженою функціональністю для складного управління запасами та гарантіями.

2. **ERP-системи для ритейлу (SAP Retail, Oracle Retail):** Чудово управляють логістикою, складами та фінансами товарів, але слабо інтегровані з телекомунікаційними системами активації послуг та білінгу.

3. **Спеціалізовані POS-системи та роздрібні CRM:** Орієнтовані на операції в точці продажу, але не "розуміють" специфіки телеком-послуг, контрактів та тарифних планів.

4. **Сучасні low-code/no-code BPM-платформи (Camunda, Microsoft Power Automate, Mendix):** Надають гнучкі інструменти для оркестрації складних бізнес-процесів, що охоплюють кілька систем. Вони можуть виступати "клеєм" між різними системами, але потребують чіткої архітектури та інтеграцій.

Висновок: На ринку відсутнє готове "коробкове" рішення, що ідеально покриває специфіку гібридної діяльності. Це обґрунтовує **необхідність проектування спеціалізованої інформаційної системи**, яка або інтегрує наявні системи в єдиний процесний ланцюг, або реалізує ключову функціональність з нуля.

1.5. Аналіз підприємства для якого виконується автоматизація

Щоб точніше сформулювати аналіз предметної області, потрібно розпочати з досліджуваних об'єктів.

Інформаційно-пошукова система може практично використовуватися в будь-якому діючому підприємстві, головним директором, замісником директора, завідувачами та керівниками, замісниками керівників структурних підрозділів, керівниками цехів, бригад – тими котрі постійно звітують перед адміністративною особою; колегіальними органами, менеджерами, котрі безпосередньо пов'язані з прийняттям управлінських рішень. Метою використання даної системи є спрощення та скорочення часових затрат на пошук відповідної інформації в існуючій базі даних.

У магазині задіяні наступні працівники: відповідальний за магазин - директор, у нього в підпорядкуванні відділ закупівель, відділ продажу, адміністратор сайту та бухгалтер. Кожен з них виконує свої специфічні функції. Відділ закупівель займається пошуком відповідного товару та його асортименту, відділ продажу займається реалізацією товару та пошуком нових клієнтів, адміністратор сайту займається рекламою в мережі Інтернет, бухгалтер веде бухгалтерський облік магазину, Директор слідкує за тим щоб діяльність фірми була налагоджена найкращим чином.

Організаційна мережа магазину мобільного зв'язку представлена на рисунку 1.1.



Рисунок 1.1 – Організаційна мережа магазину побутової техніки

Попередній аналіз предметної області свідчить про необхідність реалізації таких основних запитів до системи:

1. Каталог всіх товарів , які пропонує магазин;
2. Виявлення всіх товарів, котрі існують на складі;
3. Отримання списку товарів по коду телефону;
4. Виведення ціни товару;
5. Замовлення вибраного товару;
6. Оформлення заказу.

На рисунку 1.2 показана структура існуючої системи. Ця система своєю структурою і функціональністю не може відповідати потребам роботи сучасної фірми.

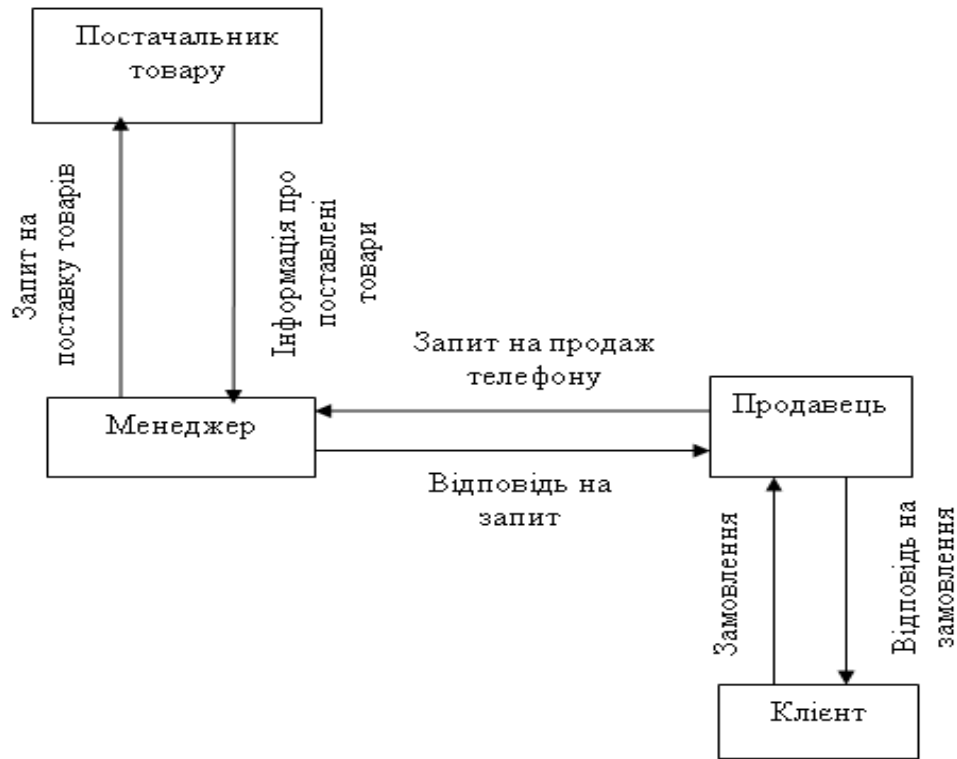


Рисунок 1.2- Структура існуючої інформаційної системи

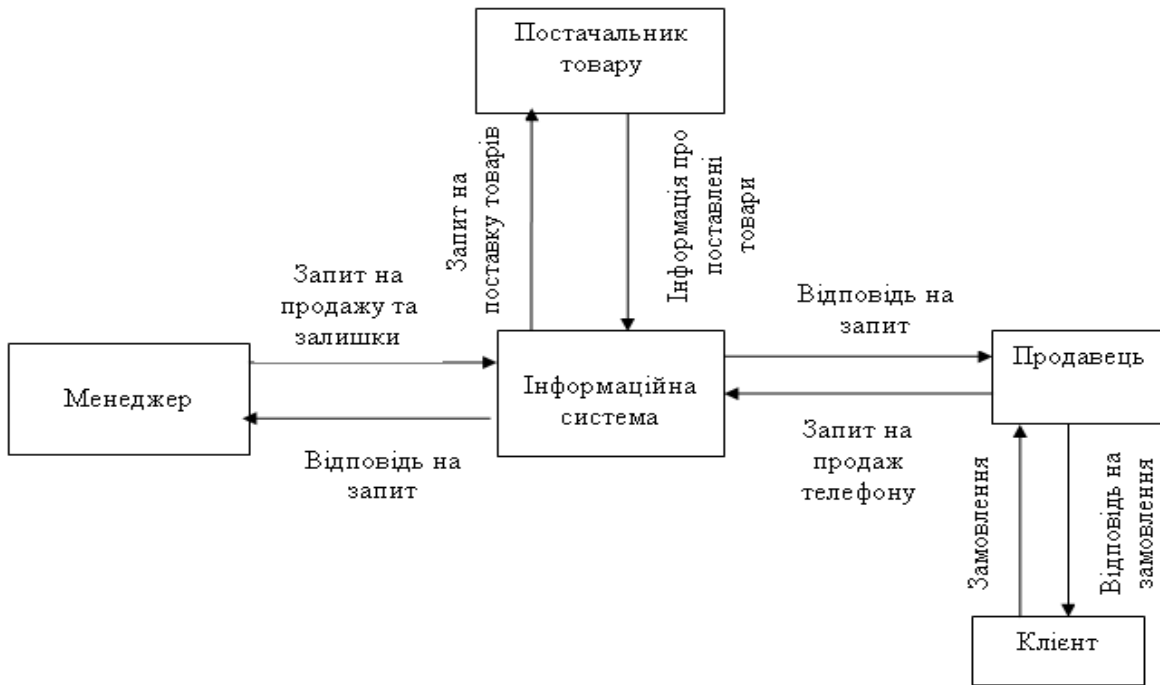


Рисунок 1.3 – Схема запропонованої системи

1) Менеджер-керуючий магазином. Обов'язками менеджера є планування, розробка і впровадження стратегічних планів для магазину, організація економічно обґрунтованої операційної діяльності, також в його обов'язки входить слідування за правильною роботою системи, розподіл права доступу до неї. Права менеджера отримувати необхідну інформацію через інформаційну систему про кількість проданого і отриманого товару.

2) Постачальник товару - це юридична або фізична особа, що поставляє товари замовникам. Постачальник здійснює підприємницьку діяльність відповідно до умов укладеного договору поставки в обумовлений строк. У даній схемі він використовуючи інформаційну систему отримує замовлення від продавця, після перевірки менеджера магазину, і виконує поставку товару згідно чинного договору.

3) Інформаційна система - є сукупність технічного, програмного, організаційного забезпечення, та призначена для того, щоб своєчасно забезпечувати належних людей належною інформацією. В даній схемі їй користуються всі учасники торгової мережі, отримуючи, аналізуючи і повертаючи необхідну інформацію.

4) Працівник – це особа яка відповідає на запит клієнта, проводить повну консультацію з даного товару і оформляє безпосередньо саме замовлення.

5) Клієнт – особа яка робить замовлення на товар, з певними вимогами до нього.

1.6. Визначення вимог до інформаційної системи продажу мобільних телефонів

На основі аналізу бізнес-процесів та технологій сформульовано базові вимоги до системи:

Функціональні вимоги:

- Єдиний каталог товарів (телефони, аксесуари) з інтеграцією з системами управління складами (WMS).
- Комплексна картка клієнта, що об'єднує історію телеком-послуг, куплених пристроїв, гарантій, фінансових угод.
- Вбудований workflow-рушій для керування процесами (BPMN-виконавець).
- Модуль фінансового супроводу з API-підключенням до банків та кредитних бюро.
- Мобільний інтерфейс для консультантів у салоні (на планшетах).
- Кабінет клієнта для відстеження статусу гарантійного ремонту, умов кредиту.
- Звіти та аналітика: конверсія по моделях телефонів, ефективність консультантів, динаміка продажів.

Нефункціональні вимоги:

- **Масштабованість:** Можливість обслуговувати сотні салонів одночасно.
- **Надійність та безпека:** Відмовостійкість, шифрування персональних та фінансових даних (відповідність GDPR/законодавству України).
- **Інтегрованість:** Відкриті API (REST, GraphQL) для інтеграції з зовнішніми системами (Billing, WMS, банки, 1C).
- **Зручність інтерфейсу (UI/UX):** Інтуїтивний та швидкий інтерфейс для консультанта, що зменшує час обслуговування.

1.7. Висновки до розділу 1

Проведений аналіз дозволив зробити такі ключові висновки:

1. Інтеграція продажу мобільних телефонів у бізнес-процеси телеком-оператора є стратегічно важливою, але технологічно складною задачею через гібридну природу операцій.
2. Існуючі процеси часто неефективні через роздробленість інформаційних систем та велику кількість ручних операцій.
3. Найбільш адекватним методологічним підходом є застосування BPM-філософії з моделюванням процесів у нотації BPMN 2.0 з акцентом на клієнтський досвід (CJM).
4. На ринку відсутнє готове рішення, що повністю задовольняє специфічні потреби, що обґрунтовує розробку спеціалізованої інформаційної системи.

РОЗДІЛ 2 РОЗРОБКА ПРОЕКТНИХ РІШЕНЬ

2.1 Інформаційне забезпечення

2.1.1 Вибір засобів моделювання

Для побудови складної інформаційної системи необхідним етапом є проектування. В останнє десятиліття в комп'ютерному світі намітилася тенденція проектування систем візуальними (наочними) моделями.

Головна особливість індустрії сучасних корпоративних програмних комплексів складається в концентрації складності на початкових етапах аналізу вимог і проектування специфікацій при відносно невисокій складності й трудомісткості наступних етапів. Фактично тут і приходить розуміння того, що буде робити майбутня програма і який чином вона буде функціонувати, щоб задовольнити пред'явленим до неї вимогам. А саме нечіткість і неповнота системних вимог, невирішені питання й помилки, допущені на етапах аналізу й проектування, породжують на наступних етапах важкі, часто нерозв'язні проблеми й, в остаточному підсумку, приводять до неуспіху всієї роботи в цілому.

Для подолання складностей початкових етапів розробки насамперед і призначений структурний аналіз - метод дослідження, що починається із загального огляду системи й потім деталізується, здобуваючи ієрархічну структуру із все більшим числом рівнів. Для таких методів характерна розбивка на рівні абстракції з обмеженням числа елементів на кожному з рівнів (звичайно від 3 до 6-7); обмежений контекст, що включає лише істотні на кожному рівні деталі; використання строгих формальних правил запису; і як наслідок - послідовне наближення до кінцевого результату.

У цей час відомо порядку 90 різновидів структурного системного аналізу, які можуть бути класифіковані стосовно шкіл (для моделювання програмних систем або систем взагалі), один по одному побудови моделі (декларуючі первинність

функціонального або інформаційного моделювання), по типі цільових систем (інформаційні системи або системи реального часу). Незважаючи на такий достаток методів, практично в усіх з них використовуються три групи засобів:

DFD (Data Flow Diagrams) - діаграми потоків даних або SADT-діаграми, що ілюструють функції, які система повинна виконувати;

ERD (Entity-Relationship Diagrams) - діаграми "сутність-зв'язок", що моделюють відносини між даними;

STD (State Transition Diagrams) - діаграми переходів станів, що моделюють залежне від часу поведіння системи (аспекти реального часу).

Крім цих моделей на етапі структурного проектування використовуються техніки структурних карт, призначені для опису відносин між модулями (структурні карти Константайна) і внутрішньої структури модулів (структурні карти Джексона).

Перераховані моделі в сукупності дають повний опис системи незалежно від того, чи є вона існуючої або знову розроблювальної.

2.1.2 Розробка діаграма варіантів використання інформаційної системи магазину продажу побутової техніки

Для зображення статичної моделі системи, уточнення функціональності, виявлення схеми функціональної структури, визначення модулів, які входять до її складу, побудуємо діаграму варіантів використання, яка відображатиме системні функції, системне оточення (діючих осіб або акторів - actors) та зв'язки між ними.

За допомогою моделі варіантів використання відобразимо інтерфейс інформаційної системи з зовнішнім світом, а саме, інформаційні потоки між системою й зовнішніми сутностями, які виникають при виконанні певних функцій. Модель варіантів використання представлена на Рисунку 2.1

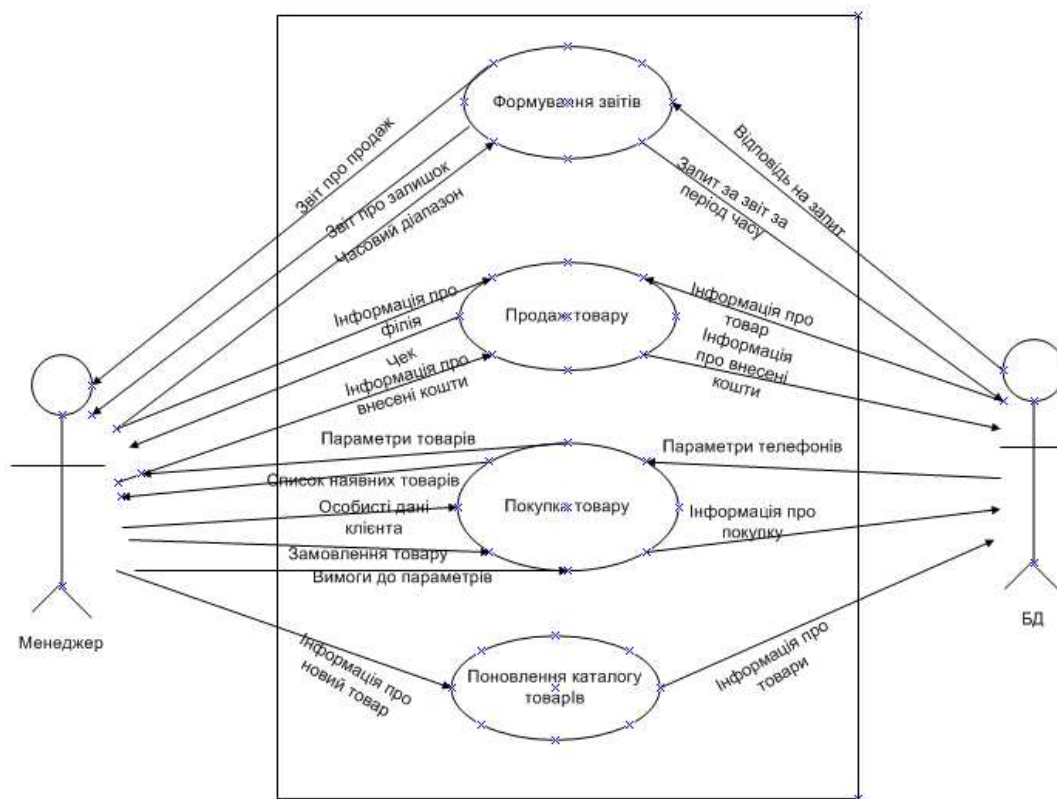


Рисунок 2.1 – Діаграма варіантів використання інформаційної систему магазину мобільного зв'язку.

Розглянемо більш детально кожний варіант використання, надаючи опис потоків подій.

2.1.3 Розробка концептуальної моделі даних

Концептуальна модель - це відображення предметної області, для якої розробляється база даних. Не вдаючись в теорію, відзначимо, що це якась діаграма з прийнятими позначеннями елементів. Так, всі об'єкти, що позначають речі, позначаються у вигляді прямокутника. Атрибути, що характеризують об'єкт - у вигляді овалу, а зв'язки між об'єктами - ромбами.

Модель сутність-зв'язок (ER-модель) - модель даних, що дозволяє описувати концептуальні схеми предметної області.

ER-модель використовується при високорівневої (концептуальному) проектуванні баз даних. З її допомогою можна виділити ключові сутності і позначити зв'язку, які можуть встановлюватися між цими сутностями.

Під час проектування баз даних відбувається перетворення ER-моделі в конкретну схему бази даних на основі вибраної моделі даних (реляційної, об'єктної, мережевий або ін.)

ER-модель являє собою формальну конструкцію, яка сама по собі не наказує ніяких графічних засобів її візуалізації. В якості стандартної графічної нотації, за допомогою якої можна візуалізувати ER-модель, була запропонована діаграма сутність-зв'язок (ER-діаграма)

Дана нотація була введена Ченом (Chen) і отримала подальший розвиток в роботах Баркера (Barker). Нотація Чена надає багатий набір засобів моделювання даних, включаючи власне ERD, а також діаграми атрибутів і діаграми декомпозиції.

Виходячи з аналізу функціональної моделі системи можна виділити конкретні сутності, ідентифікувати відношення між ними, визначити типи цих відношень. Розглянемо на прикладі відділу з продажів мобільних телефонів.

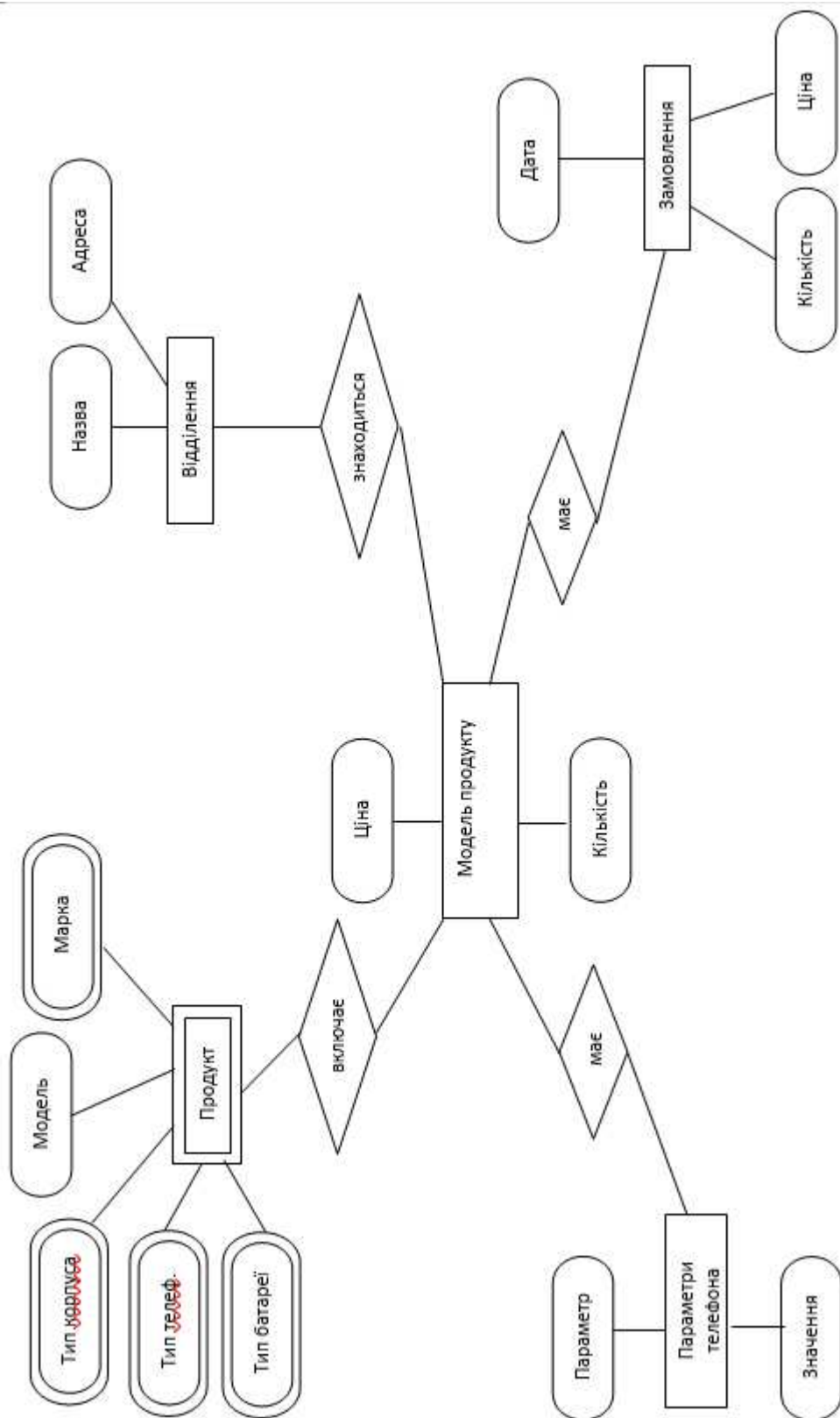


Рисунок 2.3 – Концептуальна модель даних

Можна виділити наступні сутності й атрибути:

Продукт: включає мобільний телефон. Ця сутність має наступні властивості:

- марка – марка мобільного телефону;
- модель – модель мобільного телефону;
- тип корпусу;
- тип телефонна;
- тип батареї;

Мобільний телефон: має зв'язок з сутністю Параметри, Відділення, Замовлення. Ця сутність має наступні властивості:

- ціна – ціна за 1 одиницю товару;
- кількість – кількість товару;

Відділення: додає мобільний телефон. Має зв'язок з сутністю мобільний телефон. Ця сутність має наступні властивості:

- назва – назва відділення;
- адрес – місце знаходження відділення;

Замовлення: має мобільний телефон. Має зв'язок з сутністю мобільний телефон. Ця сутність має наступні властивості:

- дата – день, число, місяць коли було прийнято замовлення;
- кількість – кількість замовлення;
- ціна – на яку суму було прийнято замовлення;

Параметри: має мобільний телефон. Має зв'язок з сутністю мобільний телефон. Ця сутність має наступні властивості:

- параметр – має значення;
- значення – має параметр.

Побудуємо концептуальну модель даних системи. Найбільш розповсюдженим засобом представлення концептуальної моделі даних є діаграми "сутність-зв'язок" (ERD). На рисунку 2.3 представлена ER діаграма у нотації Чена.

2.1.4 Логічна модель даних

Базою для побудови логічної моделі даних є концептуальна модель. Логічна модель є більш детальним представленням даних.

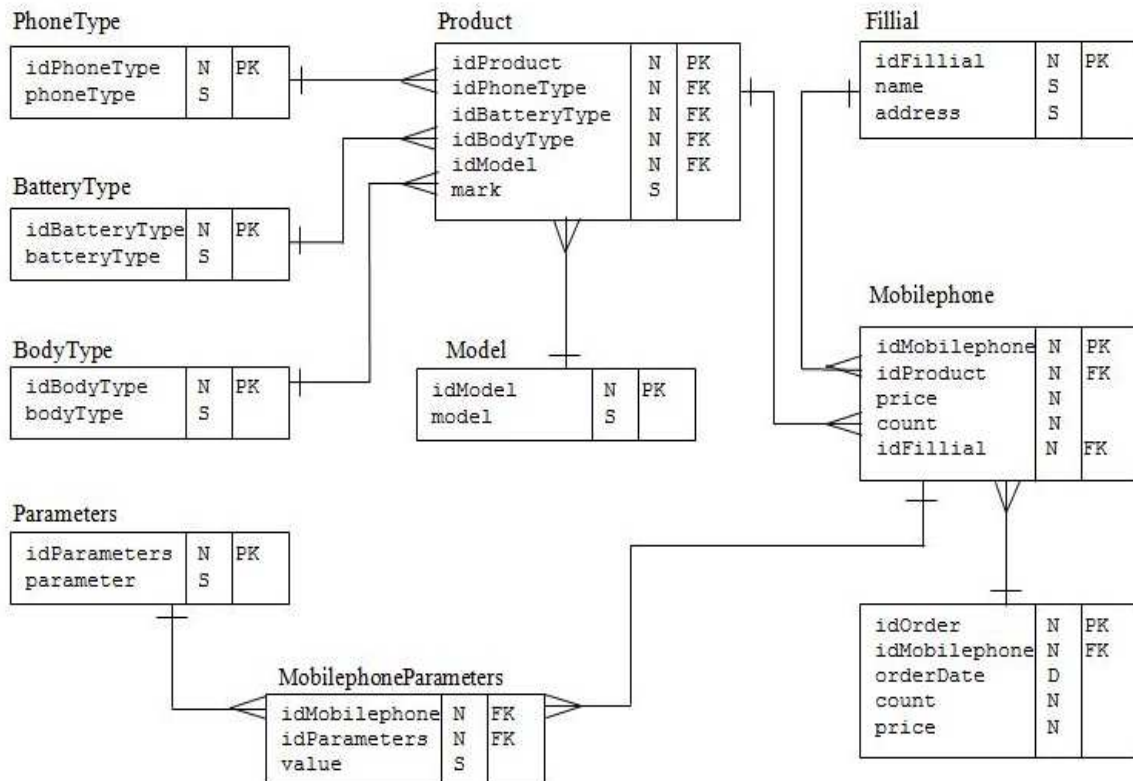


Рисунок 2.4 – Логічна модель даних

В ній сутності трансформуються в таблиці даних, одна сутність може представлятися декількома таблицями, атрибути сутностей стають полями таблиць, уточнюються відношення між таблицями, виділяються первинні ключі.

Логічні моделі структур даних повинні забезпечувати можливість реалізувати інформаційні моделі, представлені у попередньому підрозділі, в певній системі програмування. У даній роботі буде використовуватись реляційна модель даних, і відповідна логічна модель програмного забезпечення автоматизації продажу мобільних телефонів з деталізацією атрибутів показана на рисунку 2.4.

2.1.5 Розробка фізичної моделі даних

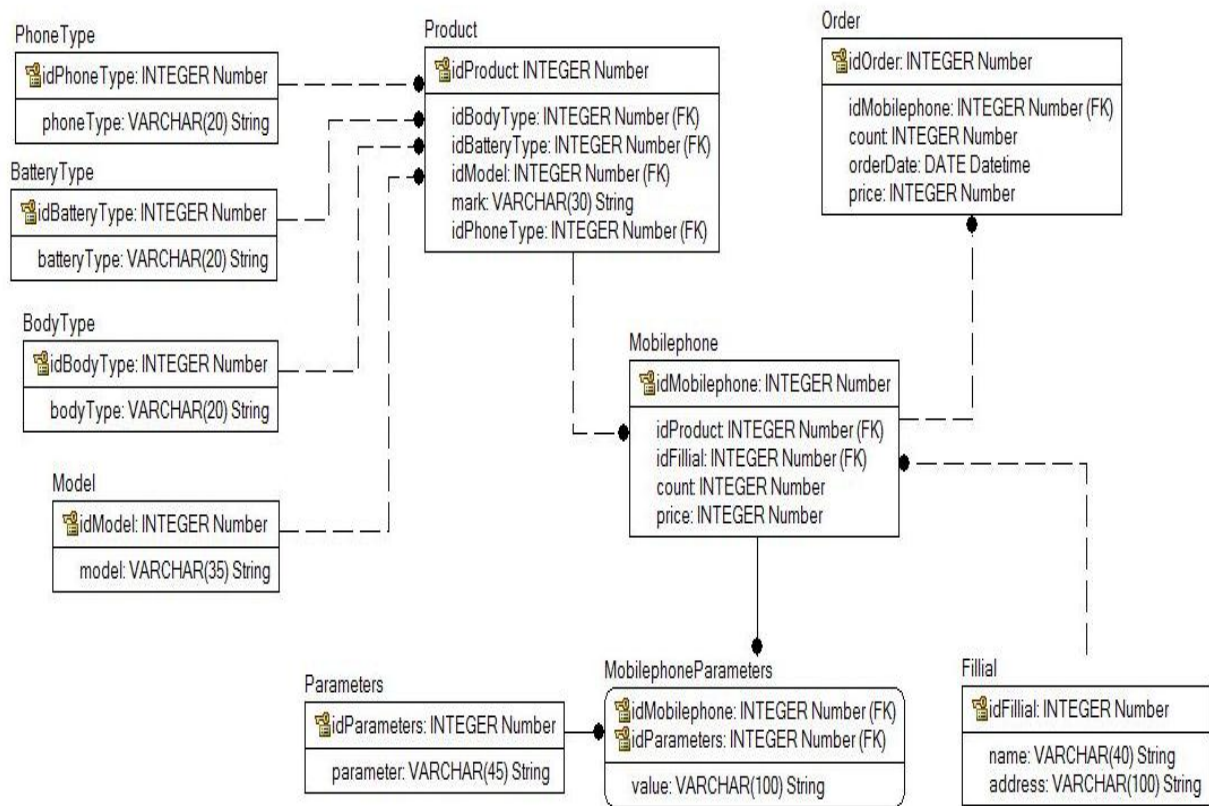


Рисунок 2.5 – Фізична модель даних

Фізична модель даних будується методом денормалізації та типізації логічної моделі даних. Розробимо фізичну структуру кожної таблиці, що входить у базу даних.

Фізична структура таблиць, які входять до БД інформаційного забезпечення представлена на рисунках 2.5.

2.2 Програмне забезпечення

2.2.1 Вибір мови програмування

Виходячи з результатів дослідження середовища виконання програмного забезпечення розробки інформаційної системи магазину мобільного зв'язку, можна сформулювати наступні вимоги до мови програмування:

- незалежність від архітектури;
- робота в мережевому середовищі;
- робота з СУБД;
- гнучкість щодо внесення майбутніх змін.

Приймаючи до уваги наведені вимоги, для вирішення поставленого завдання була обрана мова, що була розроблена програмістами Sun Microsystems, - Java.

Фірма Sun Microsystems визначає мову програмування Java наступним образом:

- об'єктно-орієнтований;
- простий;
- розподілений;
- стійкий;
- надійний;
- архітектурно-нейтральний;
- переносимий;
- інтерпретуємий;
- багатопоточний;

- динамічний.

Java орієнтований для застосування у мережі Internet/intranet має бібліотеку для роботи з протоколами TCP/IP, FTP и HTTP. Програми, які створені на мові Java, спілкуються з мережевими URL об'єктами так же просто, як з об'єктами локальної файлової системи. Сучасні версії мови Java пристосовані для створення серверних WWW програм в багатоцілонкових корпоративних системах.

Компілятор Java генерує об'єктний байт-код, якій не залежить від операційної платформи середовища виконання програми. Один й той самий байт-код може бути виконаний в різних операційних системах на різних апаратних засобах. Байт-код інтерпретується або преображається в машинний код даної платформи "на лету" за допомогою віртуальної Java машини (JVM).

Слід зазначити той факт, що мові Java уже багато років і у своєму розвитку вона перетерпіла певні зміни, що відбилися на її синтаксисі та бібліотеках.

2.2.2 Методи та засоби розробки

Для розробки діаграм була використана наступна програма ErWin 9.6.

CA ERwin Data Modeler - CASE-засіб для моделювання реляційних баз даних.

CA ERwin Process Modeler

ERwin - засіб розробки структури бази даних (БД). ERwin поєднує графічний інтерфейс Windows, інструменти для побудови ER-діаграм, редактори для створення логічного та фізичного опису моделі даних і прозору підтримку провідних реляційних СУБД і настільних баз даних. За допомогою ERwin можна створювати або проводити зворотну розробку баз даних. Процес побудови інформаційної моделі в ERwin складається з наступних кроків:

- визначення сутностей;

- визначення залежностей між сутностями;
- завдання первинних та альтернативних ключів;
- визначення атрибутів сутностей;
- приведення моделі до необхідному рівню нормальної форми;
- перехід до фізичного опису моделі: призначення відповідностей «ім'я сутності - ім'я таблиці», «атрибут сутності - атрибут таблиці»;
- завдання тригерів, процедур і обмежень;
- генерація бази даних.

ERwin створює візуальне представлення (модель даних) для розв'язуваної задачі. Це подання може використовуватися для детального аналізу, уточнення і поширення частини документації, необхідної в циклі розробки. Також ERwin автоматично створює базу даних (таблиці, індекси, збережені процедури, тригери для забезпечення посилальної цілісності та інші об'єкти, необхідні для управління даними).

Рекомендується активно використовувати можливість спадкування класів, щоб спростити розмір кінцевого коду і тим самим знизити кількість ймовірних помилок.

2.2.3 Структура програмного забезпечення.

Вхідними даними для даного етапу проектування будуть моделі сутностей, а результатом стане модель специфікації проектних конструкцій розроблювального програмного забезпечення. У відповідності з прийнятим у даної роботи підходом об'єктно-орієнтованого аналізу і проектування виконаємо подальшу специфікацію проектних рішень у виді діаграм класів.

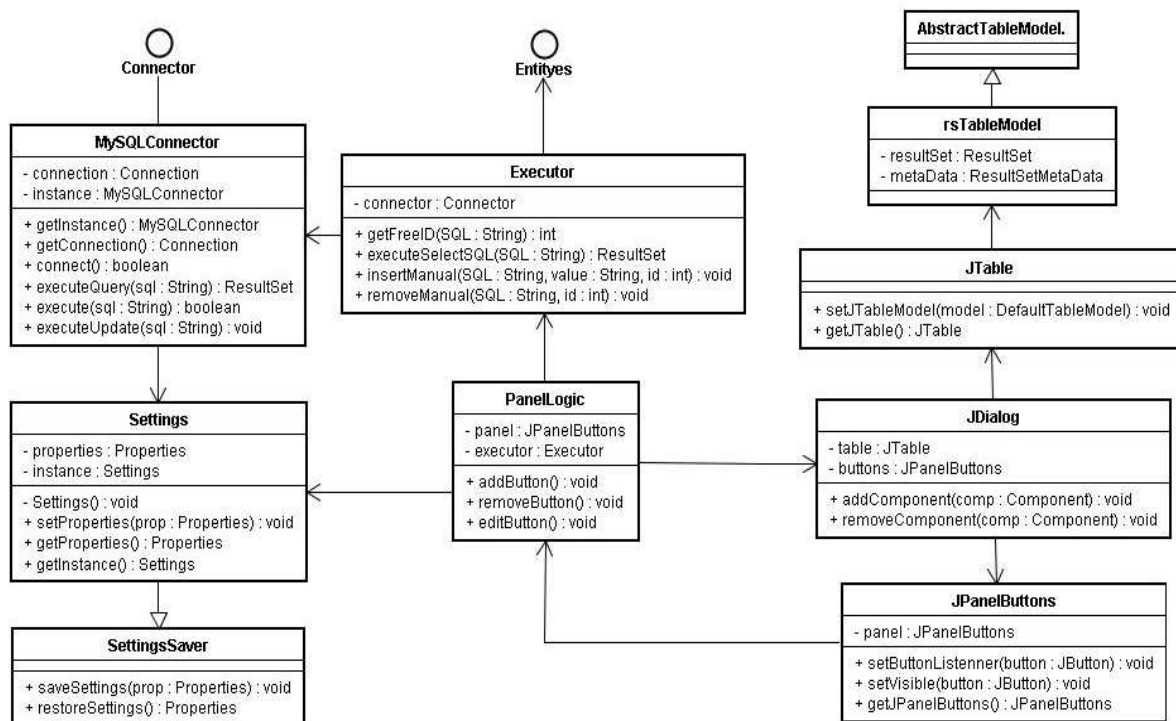


Рисунок 2.6 – Діаграма класів системи

Класи визначають стан пропонованої програмної системи (через значення своїх атрибутів) і, значною мірою, її поведінку (через операції чи методи класів). Діаграма бізнесів-сутностей приведена на рисунку 2.6.

Сформульовані користувальницькі вимоги тепер приймемо за обмеження.

Відповідальність кожного з приведених на діаграмі класів:

Connector (Інтерфейс) – для використання різних СУБД;

MySQLConnector– реалізація інтерфейса, та підключення до БД;

Entityes - інтерфейс з запитами до БД;

AbstractTableModel– абстрактна модель таблиці;

rsTableModel – успадкування і перевизначення методів абстрактної моделі таблиці;

JTable (таблиця з методами) – встановлення моделі отримання таблиці;

JDialog – вікно з таблицею і набір кнопок;

JPanelButtons – панель з кнопками;

PanelLogic (логіка програми) – методи які виконуються при натисканні на кнопки;

Executor - клас який виконує запити до БД;

SettingsSaver (зчитування) – запис налаштувань в файл;

Settings – налаштування підключення до БД (зміна, отримання).

2.2.4 Розгортання програмного забезпечення

Діаграма розгортання необхідна для візуалізації елементів та компонентів системи. Розробка діаграми розгортання являється останнім етапом специфікації моделі системи. При розробці даної діаграми визначається розподіл компонентів системи по її фізичним вузлам.

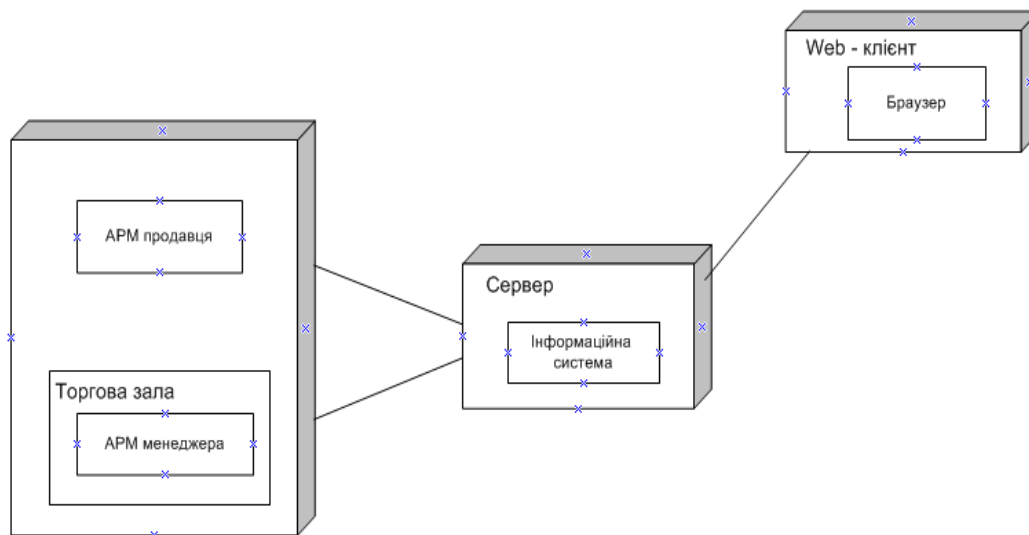


Рисунок 2.7 – Діаграма розгортання системи

Висновок до розділу 2

У процесі проєктування інформаційної системи продажу мобільних телефонів було визначено ключові вимоги до її функціональності, архітектури та інтерфейсу користувача. Проведений аналіз бізнес-процесів у сфері мобільного зв'язку дозволив сформуванати структуру системи, яка відповідає сучасним тенденціям автоматизації та забезпечує ефективну підтримку процесів обслуговування клієнтів.

Основні результати проєктування:

- сформовано модель бізнес-процесів, що охоплює управління каталогом товарів, замовленнями та клієнтською базою;
- розроблено архітектуру системи на основі клієнт–серверної технології, що забезпечує масштабованість та надійність;
- визначено структуру бази даних для зберігання інформації про товари, клієнтів та транзакції;
- спроектовано інтерфейс користувача з урахуванням принципів зручності та інтуїтивності;
- передбачено механізми захисту даних та адміністрування доступу.

Таким чином, проєктування системи стало фундаментом для її подальшої реалізації та впровадження. Запропоновані рішення дозволяють забезпечити оптимізацію бізнес-процесів продажу мобільних телефонів, підвищити ефективність роботи персоналу та покращити якість обслуговування клієнтів.

РОЗДІЛ 3 РЕАЛІЗАЦІЯ ПРОЕКТУ ІНФОРМАЦІЙНОЇ СИСТЕМИ

Розроблена інформаційна система продажу мобільних телефонів призначена для автоматизації основних бізнес-процесів торговельного підприємства, що спеціалізується на реалізації мобільних пристроїв. Система забезпечує комплексне управління процесами замовлення, обліку товарів, взаємодії з клієнтами та формування аналітичних звітів.

Основні функціональні можливості системи:

- **Управління каталогом товарів:** створення та редагування карток мобільних телефонів із зазначенням характеристик (модель, виробник, технічні параметри, ціна, наявність).
- **Облік замовлень:** реєстрація нових замовлень, відстеження їхнього статусу, формування рахунків та контроль виконання.
- **Робота з клієнтами:** збереження даних про покупців, історії їхніх замовлень, формування персоналізованих пропозицій та знижок.
- **Аналітика та звітність:** побудова звітів щодо обсягів продажів, популярності моделей, фінансових показників та ефективності роботи персоналу.
- **Інтеграція з платіжними системами:** підтримка онлайн-оплати та автоматичне підтвердження транзакцій.
- **Адміністрування:** управління правами доступу користувачів, налаштування параметрів системи, резервне копіювання даних.

Система реалізована на базі сучасних веб-технологій із використанням архітектури «клієнт–сервер». Для зберігання даних застосовується реляційна база даних, що забезпечує надійність та масштабованість. Інтерфейс користувача

розроблений з урахуванням принципів зручності та інтуїтивності, що дозволяє швидко освоїти роботу навіть новим співробітникам.

Переваги системи:

- скорочення часу обслуговування клієнтів;
- зменшення кількості помилок при обліку товарів та замовлень;
- підвищення рівня задоволеності клієнтів завдяки персоналізованим пропозиціям;
- можливість оперативного аналізу ринку та прийняття управлінських рішень;
- гнучкість у масштабуванні та адаптації під потреби підприємства.

Висновок до розділу 3

Запровадження системи дозволяє оптимізувати бізнес-процеси продажу мобільних телефонів, підвищити ефективність роботи персоналу та забезпечити конкурентні переваги на ринку. Вона може бути використана як у великих торговельних мережах, так і в невеликих компаніях, що прагнуть автоматизувати свою діяльність та покращити якість обслуговування клієнтів.

РОЗДІЛ 4. РОЗРАХУНОК ЕКОНОМІЧНОГО ЕФЕКТУ ВІД ВПРОВАДЖЕННЯ ІС

4.1 Вступ до проблеми

Інформаційні системи широко використовуються багатьма організаціями України різної форми власності. Ці системи розповсюджуються все більше і більше, змінюючись відповідно до вимог замовників та модернізуючись з плином часу.

Системи управління контентом в цілому не є принципово новою технологією, однак як парадигма створення інформаційних систем та надання послуг у сфері ІТ вони знаходяться лише на початку свого шляху. Безперечно, з розвитком цієї технології все більше і більше компаній використовуватимуть системи управління контентом для створення власної комп'ютерної інфраструктури, вирішення обчислювальних задач та ін.

Створення інформаційної системи вимагає одноразових витрат на її розробку та придбання необхідних технічних засобів, а також поточних витрат, пов'язаних з функціонуванням підсистеми. Економічний ефект від експлуатації системи визначається з урахуванням витрат на її експлуатацію. Відношення економічного ефекту до витрат на створення системи характеризує економічну ефективність капітальних вкладень. Економічні показники визначаються за діючими на момент розрахунку цінами, тарифами та ставками заробітної плати.

4.2 Розрахунок вартості створення інформаційної системи волонтерської діяльності

Загальні витрати (на проектування, реалізацію та впровадження системи)

$$З = C_{п-р} + C_{тз} + C_{пз} + C_{впр},$$

де $C_{п-р}$ – витрати на проектування і реалізацію;

$C_{\text{тз}}$ – витрати на придбання або модернізацію обчислювальної техніки, периферійних пристроїв, засобів зв'язку, допоміжного устаткування та оргтехніки (з урахуванням витрат на транспортування, монтаж, налагодження і запуск), виробничо-господарського інвентарю;

$C_{\text{пз}}$ – витрати на придбання та інсталяцію програмного забезпечення (ПЗ), необхідного для розробки;

$C_{\text{впр}}$ – витрати на впровадження системи (навчання персоналу тощо).

Витрати на проектування та реалізацію розраховуються за формулою

$$C_{\text{п-р}} = C_3 + C_{\text{від}} + C_{\text{м}} + C_{\text{н}},$$

де C_3 – заробітна плата розробників;

$C_{\text{від}}$ – відрахування на соціальні заходи (відповідно до чинного законодавства (становить 47,5% від фонду оплати праці);

$C_{\text{м}}$ – експлуатація та амортизація, $C_{\text{м}} = Z_{\text{експл}} + Z_{\text{амор}}$;

$C_{\text{н}}$ – накладні витрати у розмірі 50-150% від витрат на оплату праці (охоплюють оплату праці управлінського персоналу з нарахуваннями; оплату службових відряджень; оплату консультаційно-інформаційних послуг; оплату ремонту і техобслуговування основних фондів, крім персональних комп'ютерів (ПК)).

1) Розробка підсистеми вимагає 3 місяці (експертна оцінка, яка базується на досвіді розробки систем такого класу). Заробітна платня розробників за місяць дорівнює 12000 грн (відповідно до сучасних розцінок та попиту на спеціалістів такого профілю). Кількість розробників – одна людина.

$$C_3 = 1 * 12000 * 3 = 36000 \text{ грн.}$$

$$C_{\text{від}} = 36000 * 0,475 = 17100 \text{ грн.}$$

2) Витрати на амортизацію ЕОМ, на яких виконується розробка:

$$Z_{\text{амор}} = (C_{\text{еом}} / T_{\text{сл}}) * (T_{\text{разр}} / 12),$$

де $C_{\text{ЕОМ}} = 20000$ грн. – балансова вартість ЕОМ;

$T_{\text{сл}} = 5$ років – термін служби ЕОМ;

$T_{\text{разр}} = 0,5$ року – час, необхідний для розробки системи, виражений в місяцях, і тому ділиться на 12, щоб одержати число років.

$$Z_{\text{амор}} = (20000/5) * 0,5 = 2000 \text{ (грн.)};$$

3) витрати на експлуатацію ЕОМ, на якій виконується розробка, полягають в оплаті споживаної нею електричної енергії:

$$Z_{\text{експл}} = P_{\text{ЕОМ}} * T_{\text{разр}} * N_{\text{рд}} * N_{\text{рч}} * E_{\text{ст}},$$

де $P_{\text{ЕОМ}} = 0,4$ кВт/год – потужність ЕОМ;

$T_{\text{разр}} = 3$ місяці – тривалість розробки;

$N_{\text{рд}} = 21$ день – число робочих днів у місяці;

$N_{\text{рч}} = 8$ годин – число годин у робочому дні;

$E_{\text{ст}} = 0,9$ грн – вартість 1 кВт електроенергії

$$Z_{\text{експл}} = 0,4 * 3 * 21 * 8 * 0,9 = 181,44 \text{ (грн.)}$$

$$C_{\text{м}} = 2000 + 181,44 = 2181,44 \text{ (грн.)}$$

$$C_{\text{н}} = 36000 * 0,5 = 18000 \text{ (грн.)}$$

$$C_{\text{п-р}} = 36000 + 17100 + 2181,44 + 18000 = 73281,44 \text{ (грн.)}$$

Витрати на програмне забезпечення не включаються до даного кошторису, оскільки у функціонуванні системи передбачається застосування безкоштовного програмного забезпечення з відкритим вихідним кодом. Витрати на апаратне забезпечення становлять 18594 грн, що охоплює придбання серверного обладнання, комутаційних пристроїв та необхідних матеріалів. До складу витрат, пов'язаних із впровадженням системи, належать також витрати на навчання персоналу, монтаж і налаштування мережі, придбання устаткування, а також витрати на програмне забезпечення.

Витрати на впровадження системи приведені в таблиці 4.1.

Таблиця 4.1 – Витрати на впровадження системи.

Найменування інвестицій	Сума, грн
Вартість устаткування	18594,00
Вартість програмного забезпечення	-
Вартість підготовки кадрів	500,00
Монтаж і налаштування мереж	300,00
Разом на впровадження системи	19394,00

Таким чином, загальні витрати на проектування, реалізацію та впровадження системи складають:

$$З = C_{п-р} + C_{тз} + C_{пз} + C_{впр} = 73281,44 + 19394 = 92675,44 \text{ (грн).}$$

Витрати на створення та експлуатацію системи в перший рік експлуатації

$$З_{се} = З + З_{ер},$$

де $З_{ер}$ – експлуатаційні витрати на технічне забезпечення за рік.

$$З_{ер} = A_{об} + З_{м} + З_{е},$$

де $A_{об}$ – амортизаційні відрахування на устаткування;

$З_{м}$ – витрати на основні і допоміжні матеріали;

$З_{е}$ – витрати на електроенергію.

Амортизаційні відрахування на устаткування складають 60% балансової вартості в рік (за чинним законодавством)

$$A_{об} = 19394 * 0,6 = 11636,4 \text{ (грн.)}$$

В масштабі підприємства річні витрати на основні та допоміжні матеріали (флеш-накопичувачі, папір тощо) визначаються в розмірі 3% вартості основного устаткування

$$З_{м} = 19394 * 0,03 = 581,82 \text{ (грн.)}$$

Річний обсяг робіт ПК у нормо-годинах

$$\Phi_{м} = 253,3 * T_{с},$$

де T_C – середнє місячне завантаження устаткування (близько 5 годин); 253,3 – середня кількість робочих днів у році.

$$\Phi_M = 253,3 * 5 = 1266,5 \text{ (годин.)}$$

Витрати на електроенергію під час 1266,5 годин роботи устаткування в рік при потужності ЕОМ 0,4 кВт/год та ціні 9 грн за 1 кВт електроенергії складуть

$$З_e = 1266,5 * 9 * 0,4 = 4559,4 \text{ (грн.)}$$

Експлуатаційні витрати на технічне забезпечення за рік

$$З_{ep} = 11636,4 + 581,82 + 4559,4 = 12674,16 \text{ (грн.)}$$

Отже, у перший рік витрати на створення та експлуатацію системи складуть

$$З_{ce} = 92675,44 + 12674,16 = 105349,6 \text{ (грн.)}$$

4.3 Розрахунок економічної ефективності

Економічна ефективність використання інформаційної системи полягає у наступному:

- зменшення часу на обслуговування системи за рахунок її відмовостійкості та автоматичного масштабування;
- можливість надання користувачам інформації в оренду.

Це зменшення відбудеться за рахунок зменшення складності користування системою, збільшення надійності, стабільності та рівня безпеки. Визначимо, що це зменшення складе 50% від сучасного.

Однак, не піддається прямій грошовій оцінці зменшення кількості помилкових та необережних рішень, підвищення оперативності керування, поліпшення організації роботи та своєчасне отримання необхідної інформації про пріоритетність варіантів рішень тощо.

Використання даної системи дозволяє вивільнити 0,1 робочого часу. Так як з системою працює біля 50 користувачів, то система умовно вивільнить 5 робітників.

Визначимо пряму економічну ефективність, ґрунтуючись на тому, що впровадження системи вивільнить 5 робітників. Річна платня за рахунок вивільнення робочого часу складе

$$\Delta C_n = 5000 * 5 * 12 - 12674,16 = 287325,84 \text{ (грн.)}$$

Річний економічний ефект розраховується за наступною формулою:

$$E_{\text{год}} = \Delta C_n - E_n * k,$$

де E_n – нормативний коефіцієнт економічної ефективності капітальних вкладень у галузь і для обчислювальної техніки приймається 0,5;

k – додаткові капітальні вкладення з урахуванням витрат на проектування, створення і функціонування системи.

$$E_{\text{год}} = 287325,84 - 0,5 * 105349,6 = 234651,04 \text{ (грн.)}$$

Строк окупності підсистеми

$$T = k / \Delta C_n$$

Тобто

$$T = 105349,6 / 287325,84 = 0,37 \text{ (року)}$$

Таким чином, витрати на створення розподіленої інформаційної системи окупляться протягом 0,37 року.

Висновок до розділу 4

В результаті розрахунків витрати на створення системи обліку робочого часу (проектування, реалізація і впровадження) складуть 92675,44 грн; експлуатаційні витрати щорічно складатимуть 12674,16 грн. Загалом в перший рік витрати на модернізацію й експлуатацію складуть 105349,6 грн.

Вивільнюється 5 умовних фахівці.

Вивільнена річна зарплатня працівників складає 287325,84 грн.

Витрати на створення системи окупляться протягом 0,37 року.

За отриманими значеннями річного економічного ефекту та строку окупності можливо зробити висновок про економічну вигідність і обґрунтованість створення інформаційної системи. Слід зазначити, що підвищити економічну ефективність цієї системи можна за рахунок запропонування стороннім користувачам систему як продукт. Таким чином можна буде скоротити строк, за який окупляться витрати на створення системи, а також здобути прибуток у вигляді продажу продукту.

РОЗДІЛ 5 ОХОРОНА ПРАЦІ

Охорона праці — це система правових, соціально-економічних, організаційно-технічних, санітарно-гігієнічних та лікувально-профілактичних заходів і засобів, спрямованих на збереження життя, здоров'я та працездатності людини в процесі трудової діяльності.

Умови праці — сукупність факторів виробничого середовища і трудового процесу, які впливають на здоров'я і працездатність людини під час виконання нею професійної діяльності.

Шкідливий виробничий фактор — виробничий фактор, вплив якого на працівника може призвести до погіршення стану здоров'я або зниження працездатності.

Охорона праці базується на законодавчих, директивних та нормативно-правових документах. Під час управління охороною праці не допускаються рішення та заходи, що суперечать чинному законодавству, державним нормативним актам про охорону праці, стандартам безпеки праці, правилам та нормам охорони праці.

До основних функцій управління охороною праці належать:

- прогнозування і планування робіт, їх фінансування;
- організація та координація робіт;
- облік показників стану умов і безпеки праці;
- аналіз та оцінка стану умов і безпеки праці;
- контроль за функціонуванням системи управління охороною праці.

Основні завдання управління охороною праці:

- навчання працівників безпечним методам праці та пропаганда питань охорони праці;
- забезпечення безпеки технологічних процесів, виробничого устаткування, будівель і споруд;
- нормалізація санітарно-гігієнічних умов праці;

- забезпечення працівників засобами індивідуального захисту;
- забезпечення оптимальних режимів праці та відпочинку;
- організація лікувально-профілактичного обслуговування;
- професійний добір працівників за певними професіями;
- удосконалення нормативної бази з питань охорони праці.

Охорона праці відіграє важливу роль як суспільний чинник, оскільки жодні трудові досягнення не можуть компенсувати втраченого здоров'я чи життя. Тому вивченню питань охорони праці приділяється значна увага. Представники багатьох наук досліджують проблеми створення безпечних та нешкідливих умов праці, адже саме за таких умов людина здатна працювати високопродуктивно, створюючи матеріальний добробут суспільства.

Основоположним законодавчим документом у сфері охорони праці є Закон України «Про охорону праці» (від 14 жовтня 1992 року № 2694-XII, з подальшими змінами). Цей Закон поширюється на всі підприємства, установи та організації незалежно від форми власності та видів діяльності, на всіх громадян, які працюють або залучаються до праці на цих підприємствах. Закон визначає основні положення щодо реалізації конституційного права працівників на охорону їхнього життя і здоров'я в процесі трудової діяльності, на належні, безпечні та здорові умови праці, регулює відносини між роботодавцем і працівником з питань безпеки, гігієни праці та виробничого середовища, встановлює єдиний порядок організації охорони праці в Україні.

5.1 Аналіз потенційно небезпечних і шкідливих факторів у приміщенні відділу кадрів

Класифікація шкідливих та небезпечних виробничих факторів наведена в таблиці 4.1.

Розглянемо основні вимоги до приміщень, де встановлені комп'ютери. Залежно від орієнтації вікон рекомендується таке забарвлення стін і підлоги приміщення:

- вікна орієнтовані на південь — стіни зеленувато-блакитного або світло-блакитного кольору; підлога — зелена;
- вікна орієнтовані на північ — стіни світло-оранжевого або оранжево-жовтого кольору; підлога — червонувато-оранжева;
- вікна орієнтовані на схід — стіни жовто-зеленого кольору; підлога — зелена або червонувато-оранжева;
- вікна орієнтовані на захід — стіни жовто-зеленого або голубувато-зеленого кольору; підлога — зелена або червонувато-оранжева.

Освітлення приміщень з комп'ютерами повинно бути змішаним.

Вимоги до освітленості в приміщеннях, де встановлені комп'ютери: при виконанні зорових робіт високої точності загальна освітленість повинна становити 300 лк, комбінована — 750 лк; для робіт середньої точності — 200 і 300 лк відповідно (згідно з ДБН В.2.5-28:2018 «Природне і штучне освітлення»).

Таблиця 4.1 Класифікація шкідливих та небезпечних виробничих факторів

Види потенційно можливих небезпечних і шкідливих виробничих факторів	Наявність факторів
1. Фізичні небезпечні і шкідливі виробничі фактори:	
– шум,	+
– вібрація,	-
– підвищене виділення тепла,	-
– підвищена вологість,	-
– наявність електромагнітних полів,	+
– наявність електростатичних полів,	+
– небезпека вибуху,	-
– небезпека виникнення пожежі,	+
– небезпека ураження електрострумом,	+
– підвищена загазованість,	-
– відкрито рухаються й обертаються частини машин.	-
2. Хімічні небезпечні і шкідливі виробничі фактори:	

– токсичні речовини: пари ртуті, пари свинцю і його сполук та ін., – подразнюючі речовини: пральні засоби, кислоти, луги, роз-чинники, кислі гази, – канцерогенні речовини: сполуки алюмінію, бензин, гас, 3,4-бензапирен (міститься у вихлопних газах автомобілів).	- - -
3. Біологічні небезпечні і шкідливі виробничі фактори: – мікроби, віруси, бактерії, – комахи, – гриби, найпростіші, – інші мікроорганізми.	+ + - -
4. Психофізіологічні небезпечні і шкідливі виробничі фактори: – малорухомість роботи, – фізична утом, – нервова напруга, – напруга аналізаторів (зорового-слухового-тактильного) – напруга рук, ніг, спини	+ + + + +

5.1.1 Електробезпека

У приміщенні відділу кадрів використовуються персональні комп'ютери та інші електричні прилади, які повинні бути встановлені з дотриманням вимог Правил улаштування електроустановок (ПУЕ, затверджені наказом Міненерго України від 21.07.2017 № 476).

Вимоги електробезпеки для електронно-обчислювальних машин і персональних комп'ютерів відображені в чинних нормативних документах, зокрема ПУЕ, Правилах пожежної безпеки в Україні (НАПБ А.01.001-2014) та експлуатаційній документації виробника.

ПК, периферійні пристрої та устаткування повинні відповідати класу зони за ПУЕ, мати апаратуру захисту від короткого замикання та інших аварійних режимів.

Лінія електромережі для живлення ПК виконується як окрема групова трипровідна мережа з фазовим, нульовим робочим та нульовим захисним провідниками. Нульовий захисний провідник використовується для заземлення (занулення) електроприймачів.

У приміщенні, де експлуатується більше п'яти ПК, на доступному місці встановлюється аварійний вимикач для відключення живлення (крім освітлення). ПК підключаються до електромережі тільки через справні штепсельні з'єднання заводського виготовлення з захисним контактом.

Забороняється підключення до двопровідної мережі, використання саморобних подовжувачів, пошкодженої ізоляції, нестандартного обладнання для опалення тощо.

5.1.2 Пожежна безпека

Для приміщень з ПК застосовуються Правила пожежної безпеки в Україні (НАПБ А.01.001-2014, затверджені наказом МВС України від 30.12.2014 № 1417).

Приміщення відноситься до категорії Д за вибухопожежною небезпекою (негорючі речовини в холодному стані).

Сховища інформації розміщуються в відокремлених приміщеннях з негорючими матеріалами. Звукопоглинальне облицювання — з негорючих або важкогорючих матеріалів.

Приміщення оснащується автоматичною пожежною сигналізацією та вуглекислотними вогнегасниками (2 шт. на 20 м²).

Не рідше одного разу на квартал очищаються агрегати від пилу.

Для поліпшення пожежної безпеки в приміщенні відділу кадрів встановлена підлога з негорючих матеріалів, папір зберігається в металевих шафах, наявні вуглекислотні вогнегасники ОУ-5 та димові датчики; проводиться регулярне вологе прибирання та вентиляція.

5.1.3 Освітлення

Освітлення в приміщенні з ПК здійснюється природним і штучним відповідно до ДБН В.2.5-28:2018 «Природне і штучне освітлення». Допускається змішане освітлення.

Штучне освітлення — система загального рівномірного освітлення. Допускається комбіноване (з місцевими світильниками).

Для загального освітлення застосовуються світильники з розсіювачами та високочастотними пускорегулювальними апаратами. Забороняється використання світильників без розсіювачів.

Як джерело світла рекомендуються люмінесцентні лампи типу ЛБ. Рівень освітленості на робочому столі — 300–500 лк.

Світильники розташовуються збоку від робочого місця, паралельно лінії зору, для уникнення відблисків.

5.1.4 Захист від шуму та вібрації

Шум несприятливо впливає на організм людини. Норми шуму для приміщень з ПК — відповідно до ДСН 3.3.6.037-99 (для розумової праці — 45–55 дБА, для роботи з ПК — до 65 дБА).

Джерела шуму — вентилятори, принтери. Заходи: розміщення принтерів в окремих приміщеннях, звукоізолюючі екрани, шумопоглинальні матеріали (негорючі або важкогорючі).

Вібрація шкідлива для організму. Рівні вібрації не повинні перевищувати значення за ДСН 3.3.6.039-99.

Заходи: амортизуючі прокладки, раціональне планування.

5.1.5 Виробничий мікроклімат

Приміщення обладнується системами опалення, кондиціювання або припливно-витяжною вентиляцією.

Параметри мікроклімату на робочих місцях з ПК відповідають ДСН 3.3.6.042-99 та ГОСТ 12.1.005-88 (таблиці 4.2, 4.3).

Таблиця 4.2 – Нормовані параметри мікроклімату для приміщень з ВДТ та ПК

Пора року	Категорія робіт	Температура повітря, °С оптимальна	Відносна вологість повітря, % оптимальна	Швидкість руху повітря, м/с оптимальна
Холодна	Легка - 1а	22–24	40–60	0,1
Холодна	Легка - 1б	21–23	40–60	0,1
Тепла	Легка - 1а	23–25	40–60	0,1
Тепла	Легка - 1б	22–24	40–60	0,2

Таблиця 4.3 – Рівні іонізації повітря приміщень при роботі на ВДТ та ПК

Рівні	Кількість іонів в 1 см ³ повітря n ⁺ n ⁻
Мінімально необхідні	400 600
Оптимальні	1500–3000 3000–5000
Максимально допустимі	50000 50000

Для підтримки норм — установки зволоження, іонізації або кондиціонування.

5.1.6 Організація робочого місця

Робоче місце — місце постійного або тимчасового перебування працівника в процесі трудової діяльності. Організація робочого місця з ПК повинна відповідати ергономічним вимогам.

Площа на одне робоче місце з ПК — не менше 6 м², об'єм — не менше 20 м³.

Робочі місця розміщуються так, щоб природне світло падало збоку (переважно зліва).

Вимоги до розміщення:

- відстань від стін зі вікнами — не менше 1 м;
- між бічними поверхнями моніторів — не менше 1,2 м;
- між тильною поверхнею одного монітора та екраном іншого — не

менше 2,5 м;

- прохід між рядами — не менше 1 м.

Висота робочої поверхні столу — 680–800 мм. Робочий стілець — підйомно-поворотний, регульований за висотою, нахилом сидіння та спинки, з підлокітниками.

Екран монітора — на відстані не ближче 600 мм від очей. Клавіатура — на відстані 100–300 мм від краю столу, з нахилом 5–15°.

5.2 Розрахунок освітлення

Розрахунок КПО попереднім методом для виробничого приміщення з боковим одностороннім освітленням. Приміщення розташоване в 4 поясі світлового клімату. Довжина приміщення: 13м, глибина: 8м. Розрахункова точка А знаходиться від зовнішньої стіни на відстані: 7 м. Відстань від робочої поверхні до верха вікон: 4.4 м.

Нормоване значення коефіцієнту бокового природного освітлення для III розряду зорової роботи III світлового поясу $e_n^{III} = 1.5\%$. У зв'язку з тим, що приміщення знаходиться в IV світловому поясі, КПО буде визначено за формулою:

$$e_n^{IV} = e_n^{III} m c \quad (4.2.1)$$

де $m = 0,9$ —коефіцієнт світлового клімату;

$C = 0,7$ - коефіцієнт сонячності клімату.

$$e_n^{IV} = 1.5 \times 0.9 \times 0.7 = 0.945\%$$

Загальний коефіцієнт світло пропускання визначається за формулою:

$$\tau_0 = \tau_1 \tau_2 \tau_3 \tau_4 \tau_5 \quad (4.2.2)$$

де $\tau_1 = 0,8$ - засклення подвійне;

$\tau_2 = 0,65$ - Плетення подвійне дерев'яне;

$\tau_3 = 1$ - Несущі конструкції відсутні;

$\tau_4 = 1$ - Сонце захисні конструкції відсутні

$\tau_5 = 1$ - Захисних сіток немає

$$\tau_0 = 0.8 \times 0.65 \times 1 \times 1 \times 1 = 0,52$$

Площа світлових прорізів при боковому освітленні визначається за формулою:

$$S_{\sigma} = \frac{S_{\pi} e_{\pi} K_3 \eta_{\sigma} K_{\sigma}}{100 r_1 \tau_0} \quad (4.2.3)$$

де S_{π} - площа підлоги приміщення,

$$S_{\pi} = 13 \times 8 = 104 \text{ м}^2;$$

$K_3 = 1,2$ - коефіцієнт запасу;

$\eta_{\sigma} = 13$ - світлова характеристика вікон;

$K_{\sigma} = 1$ - коефіцієнт, що враховує затінення вікон будинками, що стоять навпроти;

$r_1 = 2,4$ - коефіцієнт, що враховує підвищення КПО при боковому освітленні завдяки світлу, відбитому від поверхонь приміщення і підстильного шару, що прилягає до будинку.

$$S_{\sigma} = \frac{104 \times 1,26 \times 1,2 \times 13 \times 1}{100 \times 2,4 \times 0,52} = 16,38 \text{ м}^2$$

Загальна ширина вікон знаходиться за допомогою наступної формули:

$$b_{\sigma} = \frac{S_{\sigma}}{h_{\sigma}} \quad (4.2.4)$$

де $h_{\sigma} = 4 \text{ м}$ - висота вікна

$$b_{\sigma} = \frac{8,24}{4} = 2,06 \text{ м}$$

Приймається 4 вікна, які мають ширину 1 м.

Висновок до розділу 5

У цьому розділі проведено аналіз потенційно небезпечних та шкідливих факторів у приміщенні відділу кадрів. Розроблено заходи щодо усунення або зниження їхнього впливу до нормативних вимог. Дотримання оптимальної організації робочого місця дозволить зберегти працездатність протягом робочого дня та підвищити продуктивність праці.

РОЗДІЛ 6. ОХОРОНА НАВКОЛИШНЬОГО СЕРЕДОВИЩА

6.1 Вступ до проблеми

Охорона навколишнього середовища здійснюється через різні засоби, зокрема й правові механізми. У сфері права захист охоплює майже всі складові природного середовища. Основними сучасними нормативно-правовими актами, які визначають основи організації охорони природи в Україні, є Закони «Про охорону навколишнього природного середовища» (25 червня 1991 р.), «Про охорону атмосферного повітря» (16 жовтня 1992 р.), «Про природно-заповідний фонд України» (16 червня 1992 р.), «Про тваринний світ» (3 березня 1993 р.), «Про карантин рослин» (30 червня 1993 р.) тощо. Крім того, питання використання і захисту природи регулюють також кодекси – земельний, водний, лісовий, про надра, а також закони «Про плату за землю» (3 липня 1992 р.), «Про ветеринарну медицину» (25 червня 1992 р.). Важливим є й затверджений Верховною Радою Порядок щодо обмеження, тимчасової заборони чи припинення діяльності підприємств, установ, організацій та об'єктів у разі порушень законодавства про охорону довкілля.

Види права природокористування включають землекористування, водокористування, лісокористування, користування надрами, тваринним світом та природно-заповідним фондом. Природокористування — це раціональне застосування природних ресурсів людиною для задоволення своїх потреб. Ключові принципи цього права: цільове використання, плановість, тривалість, ліцензування, а також врахування його особливої ролі для суспільства. Виокремлюють загальне й спеціальне природокористування стосовно землі, вод, лісу, надр, тваринного світу тощо. Згідно із Законом «Про охорону навколишнього природного середовища», усі громадяни мають право на загальне користування природними ресурсами безкоштовно та без спеціальних дозволів. Це стосується відвідування парків, скверів, водойм, лісу, збору дикорослих ягід, грибів, горіхів тощо. Стаття

13 Конституції України гарантує це право кожному. Спеціальне природокористування виникає, коли частину ресурсів держава передає у визначених законом випадках індивідам або організаціям на певний термін і за відповідну плату; передача закріплюється спеціальним дозволом.

Права природокористувачів супроводжуються екологічними правами та обов'язками, що інтенсивно розвиваються в юридичній науці. За Конституцією України, кожен має право на безпечне довкілля і відшкодування шкоди, завданої порушенням цього права. Також гарантується доступ до інформації про стан довкілля та її поширення, що закріплено і в Законі «Про охорону навколишнього природного середовища». Над цим правом лежить обов'язок держави впроваджувати санітарно-гігієнічні заходи для покращення стану довкілля. Екологічні права громадян захищаються у судовому порядку. Разом із цим Закон передбачає певні обов'язки, незалежно від статусу природокористувача: усі зобов'язані берегти природу, використовувати ресурси раціонально та не завдавати їм шкоди. Для суб'єктів спеціального використання природних ресурсів встановлюються додаткові обов'язки, включно зі сплатою за використання та забруднення довкілля згідно з нормативами і лімітами. Контроль у цій сфері здійснюють як державні органи, так і громадські організації шляхом перевірок, нагляду, обстежень та експертиз. Державний контроль здійснюють Ради народних депутатів, державні адміністрації та Міністерство охорони навколишнього природного середовища разом із його місцевими структурами.

Основні пріоритети охорони довкілля та раціонального використання природних ресурсів включають:

- забезпечення екологічної безпеки ядерних об'єктів і захист населення й довкілля від радіації, а також мінімізацію негативних наслідків аварії на Чорнобильській АЕС;
- покращення екологічного стану річкових басейнів України та якості питної

води;

- стабілізацію й поліпшення екологічної ситуації у містах і промислових центрах Донецько-Придніпровського регіону;
- будівництво та модернізацію міських очисних каналізаційних споруд;
- боротьбу із забрудненням Чорного й Азовського морів та покращення їхнього стану;
- формування збалансованої системи природокористування, структурну перебудову економіки, екологізацію процесів у промисловості, енергетиці, будівництві, сільському господарстві та транспорті;
- збереження біологічного й ландшафтного різноманіття, розвиток заповідної справи.

Для досягнення цих цілей передбачено такі завдання:

- зменшення рівня радіаційного забруднення до мінімуму;
- захист повітря від забруднення, особливо у великих містах і промцентрах;
- захист і збереження земель від виснаження, забруднення та нераціонального використання;
- збереження та розширення природних ландшафтів, посилення охорони заповідних і рекреаційних територій;
- підвищення стійкості лісів і їхніх екологічних функцій;
- знешкодження, утилізація та захоронення побутових і промислових відходів;
- запобігання забрудненню вод, скорочення й припинення скиду забруднених стоків у водойми, захист підземних вод;
- збереження й відродження малих річок, управління водними ресурсами за басейновим принципом;
- завершення формування державної системи моніторингу довкілля;
- створення систем прогнозування, запобігання й оперативного реагування на

надзвичайні ситуації природного та техногенного характеру;

- екологічний супровід конверсії військово-промислового комплексу;
- контроль за екологічною діяльністю Збройних Сил України;
- розробка ефективних механізмів природокористування;
- впровадження економічних важелів впливу на систему природокористування;
- формування системи екологічної освіти, виховання й інформування.

6.2 Забруднення навколишнього середовища підприємством з розробки програмного забезпечення

Основним джерелом забруднення навколишнього середовища підприємством є використання автотранспорту. Підприємство впливає на стан атмосферного повітря, однак не спричиняє забруднення водного чи ґрунтового середовища.

У сучасних умовах автомобільний транспорт є ключовим фактором погіршення якості атмосферного повітря, особливо в межах великих міст. Атмосфера урбанізованих територій характеризується постійним надходженням забруднюючих речовин, що істотно відрізняє її параметри від природного стану. У міських поселеннях спостерігаються найвищі рівні антропогенних навантажень, унаслідок чого змінюються мікрокліматичні фактори та фізико-хімічні властивості повітряного басейну. Згідно з метеорологічними дослідженнями, температура повітря у містах переважно вища, ніж у передмісті, а також частіше фіксуються атмосферні опади, тумани та смоги – густі утворення, що містять дим, кіптяву і вихлопні гази. Прозорість міської атмосфери є нижчою порівняно із сільською місцевістю, а запиленість та тумани суттєво зменшують проникнення сонячної радіації.

Якість атмосферного повітря у містах дедалі більше погіршується через викиди автотранспортних засобів. Збільшення кількості автомобілів призводить до

зростання об'ємів забруднювачів, зокрема оксидів вуглецю, вуглеводнів, оксидів азоту, сажі, а також свинцевих і сірчаних сполук, концентрація яких значно підвищилась. Міська інфраструктура не розрахована на існуючу кількість автотранспорту, що спричиняє короткі ділянки безперервного руху (400–600 м) та низьку середню швидкість (12–20 км/год), збільшуючи витрати пального у 3–4 рази, відповідно посилюючи викиди шкідливих речовин.

Автотранспорт також є джерелом специфічних видів забруднення, зокрема пилу від стирання шин, який у значних обсягах потрапляє у повітря. Окрім продуктів згоряння пального, автомобільний транспорт сприяє накопиченню свинцю в довкіллі. В Україні продовжують використовувати бензин із вмістом свинцю 0,36 г/л, тоді як у Великій Британії, Німеччині та США цей показник становить 0,013–0,15 г/л.

6.3 Розробка заходів щодо зменшення забруднення

Природоохоронна діяльність охоплює всі заходи, спрямовані на підтримку якості довкілля на рівні, що гарантує стійкість біосфери. Вона включає масштабні державні програми зі збереження унікальних природних територій та різноманіття видів, організацію наукових досліджень, підготовку екологічних фахівців і просвітницьку роботу серед населення. Крім того, підприємства здійснюють локальні дії: очищають стічні води та відхідні гази, знижують використання природних ресурсів тощо. Здебільшого такі заходи реалізуються інженерними методами.

Виділяють два основні напрями природоохоронної діяльності підприємств. Перший — очищення шкідливих викидів, який сам по собі є малоефективним, оскільки не завжди дозволяє повністю припинити надходження забруднюючих речовин у біосферу. Зменшення забруднення одного компонента часто призводить до посилення забруднення іншого. Наприклад, впровадження вологих фільтрів у

системах газоочистки зменшує забруднення повітря, але збільшує забруднення води. Уловлені речовини можуть отруювати значні площі землі. Очищувальні споруди скорочують рівень забруднення, проте цілком проблему не вирішують, адже самі генерують нові, хоча й менш об'ємні, але концентровані відходи. Діяльність більшості очисних установок потребує значних енергетичних ресурсів, створюючи додатковий екологічний ризик.

Значна частина забруднювачів, які витрачають кошти на їхнє знешкодження, є речовинами, що вже мають виробничу цінність і можуть бути використані у господарстві. Для інтеграції еколого-економічних рішень важливо поєднати процес очищення шкідливих викидів із утилізацією уловлених речовин.

Другий напрям — усунення причин забруднення шляхом розробки маловідходних або безвідходних технологій, які дають змогу раціонально використовувати сировину і максимально утилізувати шкідливі речовини. Однак зараз не для всіх галузей знайдено економічно прийнятні рішення щодо зменшення відходів та їх утилізації, тому необхідно діяти комплексно у двох напрямках.

Слід пам'ятати, що жодні очисні споруди чи безвідходні технології не здатні відновити стійкість біосфери, якщо буде перевищено граничні показники перетворення природних систем, згідно із законом незамінності біосфери. Критичними є використання понад 1% енергії біосфери й трансформація понад 10% природних територій (правило одного і десяти відсотків). Тому технічний прогрес не замінює необхідності перегляду суспільних пріоритетів, стабілізації населення, створення заповідників та інших заходів.

Основні напрями захисту атмосфери від автотранспортних викидів:

- розвиток і виробництво автомобілів із високоефективними, малотоксичними двигунами, включаючи дизелізацію транспорту;
- впровадження дієвих систем нейтралізації вихлопних газів;
- зниження токсичності моторних палив;

- оптимізація організації транспортного руху в містах та вдосконалення дорожнього будівництва для забезпечення безперервного руху на автомагістралях.

Висновок до розділу 6

Отже, забезпечення ефективного захисту навколишнього середовища вимагає інтегрованого підходу, який поєднує впровадження технологічних інновацій із цілеспрямованою трансформацією суспільних цінностей. Необхідно не лише застосовувати сучасні методи очищення та нейтралізації забруднень, а й оптимізувати використання природних ресурсів і сприяти збереженню екосистем. Лише всебічне поєднання технологічних рішень і свідомої екологічної відповідальності дозволить підтримати стабільність біосфери для майбутніх поколінь.

ВИСНОВКИ

У ході дослідження було проаналізовано сучасні методи та технології управління бізнес-процесами у сфері мобільного зв'язку. Встановлено, що ефективність роботи компаній значною мірою залежить від рівня автоматизації процесів обслуговування клієнтів та інтеграції інформаційних систем.

Виявлено ключові проблеми у сфері управління бізнес-процесами: недостатня гнучкість традиційних моделей, складність адаптації до швидких змін ринку мобільного зв'язку, а також потреба у персоналізації сервісів для клієнтів.

Проведений аналіз показав, що використання сучасних BPM-технологій (Business Process Management) дозволяє оптимізувати процеси взаємодії з клієнтами, скоротити час обслуговування та підвищити якість сервісу.

У роботі розроблено концепцію та реалізовано інформаційну систему продажу мобільних телефонів, яка забезпечує:

- автоматизацію процесів оформлення та супроводу замовлень;
- інтеграцію з базами даних клієнтів та товарів;
- можливість аналітики продажів та прогнозування попиту;
- підвищення рівня клієнтського сервісу завдяки швидкому доступу до інформації.

Практична цінність роботи полягає у створенні інструменту, що може бути використаний компаніями мобільного зв'язку для оптимізації бізнес-процесів та підвищення конкурентоспроможності на ринку.

Результати дослідження підтверджують, що впровадження інформаційних систем у сфері мобільного зв'язку сприяє не лише підвищенню ефективності управління, але й формуванню довгострокових відносин із клієнтами, що є стратегічно важливим для розвитку галузі.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Spring Boot. URL: <https://spring.io/projects/spring-boot> (дата звернення: 20.05.2025).
2. PostgreSQL. URL: <https://www.postgresql.org/> (дата звернення: 20.05.2025).
3. Hibernate. URL: <https://hibernate.org/> (дата звернення: 20.05.2025).
4. Spring MVC. URL: <https://docs.spring.io/spring-framework/docs/current/reference/html/web.html> (дата звернення: 20.05.2025).
5. Thymeleaf. URL: <https://www.thymeleaf.org/> (дата звернення: 20.05.2025).
6. Bootstrap. URL: <https://getbootstrap.com/> (дата звернення: 20.05.2025).
7. Fowler M. Microservices. URL: <https://martinfowler.com/articles/microservices.html> (дата звернення: 20.05.2025).
8. Fielding R. T. Architectural Styles and the Design of Network-based Software Architectures. Doctoral dissertation. University of California, Irvine, 2000.
9. Spring Security. URL: <https://spring.io/projects/spring-security> (дата звернення: 20.05.2025).
10. JSON Web Tokens. URL: <https://jwt.io/> (дата звернення: 20.05.2025).
11. JUnit. URL: <https://junit.org/> (дата звернення: 20.05.2025).
12. Mockito. URL: <https://site.mockito.org/> (дата звернення: 20.05.2025).
13. Apache JMeter. URL: <https://jmeter.apache.org/> (дата звернення: 20.05.2025).
14. Моделі в програмній інженерії: навч. посібник / В. В. Яковина, Д. С. Федасюк, О. О. Нитребич, І. І. Ковалюк. Львів: Львівська політехніка, 2020. 196 с.
15. Patterns of Enterprise Application Architecture / М. Fowler. - Addison-Wesley Professional, 2002. - 560 p.
16. Pro Spring 5: An In-Depth Guide to the Spring Framework and Its Tools / І. Cosmina et al. - Apress, 2017. - 868 p.

17. Clean Code: A Handbook of Agile Software Craftsmanship / R. C. Martin. - Pearson, 2008. - 464 p.
18. Test-Driven Development: By Example / K. Beck. - Addison-Wesley Professional, 2002. - 240 p.
19. Spring Start Here: Learn what you need and learn it well / L. Spilca. - Manning Publications, 2021. - 528 p.
20. Learning Web Design: A Beginner's Guide to HTML, CSS, JavaScript, and Web Graphics / J. Niederst Robbins. - O'Reilly Media, 2018. - 808 p.
21. CSS: The Definitive Guide: Visual Presentation for the Web / E. A. Meyer, E. Weyl. - O'Reilly Media, 2017. - 1090 p.
22. JavaScript: The Definitive Guide: Master the World's Most-Used Programming Language / D. Flanagan. - O'Reilly Media, 2020. - 706 p.
23. Наказ Міністерства соціальної політики України «Про затвердження Вимог щодо безпеки та захисту здоров'я працівників під час роботи з екранними пристроями» від 14.02.2018 № 207. URL: <https://zakon.rada.gov.ua/laws/show/z0508-18> (дата звернення: 14.05.2025).
24. ДСанПіН 3.3.2-007-98 «Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин». URL: <https://zakon.rada.gov.ua/rada/show/v0007282-98> (дата звернення: 14.05.2025).
25. ДСТУ ISO 9241-5:2004. Ергономічні вимоги до роботи з відеотерміналами в офісі. Частина 5. Вимоги до компонування робочого місця та до робочої пози. [Чинний від 2006-01-01]. Вид. офіц. Київ, 2006. 24 с.
26. Вимоги до організації робочого місця користувача комп'ютера: веб-сайт. URL: <https://zp.dsp.gov.ua/articles/a-657.html> (дата звернення: 14.05.2025).
27. Охорона праці користувачів комп'ютерів: веб-сайт. URL: https://pidru4niki.com/1821100138196/bzhd/ohorona_pratsi_koristuvachiv_kompyuteri (дата звернення: 14.05.2025).

28. ДСанПіН 3.3.2-007-98 «Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин». URL: <https://zakon.rada.gov.ua/rada/show/v0007282-98> (дата звернення: 14.05.2025).

29. Жидецький В.Ц. Основи охорони праці. Підручник. [Текст] / Жидецький В.Ц. Львів: Афіша, 2002. – 320 с.

30. Жидецький В.Ц. Охорона праці користувачів комп'ютерів . Навчальний посібник. – Вид.2-ге доп. [Текст] / Жидецький В.Ц Львів: Афіша, 2000. – 176 с.

31. ДНАОП 0.00 – 1.31 – 99. Правила охорони праці під час експлуатації електронно-обчислювальних машин. [Текст] / К., 1999. – 111 с.

32. М.П. Купчик Охорона праці. Лабораторний практикум [Текст] / М.П. Купчик, М.П. Гандзюк, І.Ф. Гандзюк, І.Ф.Степанєв, В.Н Вендиханський, А.М. Литвиненко, О.В Іваненко. – К.: Основа, 1998. - 224 с.

33. Тубальцев А.М. Виробниче освітлення та його розрахунок: навчальний посібник. . [Текст] / Тубальцев А.М. –Миколаїв: УДМТУ, 2001. – 84 с.

34. Щедролюєв О.В. Основи охорони праці. Лабораторний практикум. [Текст] / Щедролюєв О.В., Моїсєєнко Л.Л., Яглицький Ю.К. – Херсон:Айлант, 2005. – 84.

ДОДАТОК А – ТЕХНІЧНЕ ЗАВДАННЯ

Призначення розробки системи

Назва програмного продукту: “Інформаційна система обслуговування клієнтів у сфері мобільного зв’язку”. Програмний продукт призначений автоматизації товаро–фінансових робіт підприємства.

Експлуатаційне призначення: при покупці товару, покупець повинен зареєструватися. Реєстрація проходить шляхом занесення особистих даних (ПІБ, телефон, адреса) в анкету з метою встановлення гарантійного строку для телефону, що купується. В даній системі автоматизуються наступні процеси: прийняти замовлення, оформити замовлення, закупка товару, продажу товару.

Мета розробки системи

Метою даної роботи є проектування такої інформаційної системи, яка б дозволила не тільки автоматизовано вести документацію і рух товарів, але й проводити обробку та аналіз замовлень з подальшим наданням звітності у різних розрізах.

Вимоги до структури і функціонування системи:

Програма повинна виконувати наступні дії:

- накопичувати інформацію про постачальників товару(назва, організація, адреса);
- зберігати інформації про клієнтів(ПІБ, адреса, код клієнта, телефон);
- зберігати інформацію про робітників(ПІБ, адреса, телефон, № паспорта, дата найма, заробітна плата);
- зберігати інформацію про товари(код,назва, № товару, кількість, марка, ціна);
- відсортувати параметри товарів по їхніх марках(код, назва марки);
- роздруковувати результат пошуку(назва марки, № товару, ціна, наявність товару)

- при наявності помилок користувача повідомляти його про цьому(помилка та заходи уникнення).

Вхідні дані вводяться працівником фірми із установленими правилами уведення даних у програмі (тип, довжина, коректність).

Вимоги до надійності

Система повинна бути надійною і коректна реагувати на можливі нестандартні ситуації. Від її роботи залежить робота всього магазину, тому її надійність повинна бути дуже висока. Для того щоб оцінити надійність програми необхідно проаналізувати можливі аварійні ситуації:

– відключення електроживлення – для запобігання цієї ситуації встановлюється джерело безперебійного живлення, що дозволяє коректно закінчити роботу;

– вихід з ладу комп'ютера – для запобігання даної ситуації необхідно проводити систематичне тестування і перевірку працездатності;

– вихід з ладу локальної мережі – для запобігання цього необхідно періодично робити тестування функціонування мережі і не допускати повного виходу з ладу.

З розглянутого вище видно, що надійність системи навіть в аварійних ситуаціях залишається дуже велика.

Вимоги до безпеки

При роботі із системою, необхідно дотримувати правила техніки безпеки при роботі з ЕОМ. До роботи із системою допускаються тільки користувачі, що ознайомлені з посібником користувача.

Вимога до експлуатації, технічному обслуговуванню, ремонтіві і збереженню

До роботи допускаються тільки підготовлені користувачі, що пройшли інструктаж з техніки безпеки роботи з ЕОМ і ознайомлені з посібником користувача. При виникненні збоїв у роботі, користувач повинний сповістити про це адміністраторові системи.

Технічне обслуговування і ремонт устаткування здійснюється тільки кваліфікованими фахівцями.

Вимоги до захисту інформації від несанкціонованого доступу

Система повинна забезпечувати доступ тільки тим користувачам, що мають відповідні права доступу, а також вести схований облік усіх змін інформації.

Вимоги по схоронності інформації при аваріях

Повинна бути забезпечена збереження інформації при наступних аварійних ситуаціях:

- 1) відключення електроживлення;
- 2) вихід з ладу серверної станції;
- 3) вихід з ладу локальної комп'ютерної мережі.

Вимоги до захисту від зовнішніх впливів

Для безпечної роботи системи необхідно дотримувати правила техніки безпеки при роботі з ЕОМ і пожежної безпеки в робочому приміщенні.

Порядок контролю і приймання системи

Для контролю й прийому повинно бути надано керівництву оператори та опис програми, а також програма й методика випробування. Порядок контролю й прийому даної розробки здійснюється представником замовника в присутності представника

розробника згідно з програмою та методикою випробувань. Якщо програма не пройшла випробування, виконавець зобов'язаний виправити помилки та недоліки в програмі, не більш ніж за 1 місяць із дня випробування.

За результатами прийому складається акт, який підписується представником замовника та представником розробника та утверджується керівниками організації-замовника й організації-розробника.

У випадку виявлення помилок під час прийому програмного виробу складається акт про виявлені помилки, який підписується представниками замовника та розробника й стверджується керівниками організації - замовника й організації - розробника.

Розробник повинен у продовж не більш як 1-го місяця виправити вказані зауваження, й повідомити замовника про повторне проведення перевірки, не пізніше як за 2 тижня до закінчення прийому програмного виробу.

Вимоги до документації

До складу програмної документації повинні входити наступні документи:

- загальний опис системи;
- методика іспитів програмного забезпечення;
- посібник користувача.
- інструкція з формування і ведення бази даних.

ДОДАТОК Б – ІНСТРУКЦІЯ КОРИСТУВАЧА

Запускаємо програму на робочому столі, після цього потрібно ввести пароль для визначення користувача. Навігаційні кнопки та панелі достатньо оснащені коментарями для розуміння та комфортної роботи з програмою.

На головному вікні зображено список заявок та клієнтів.

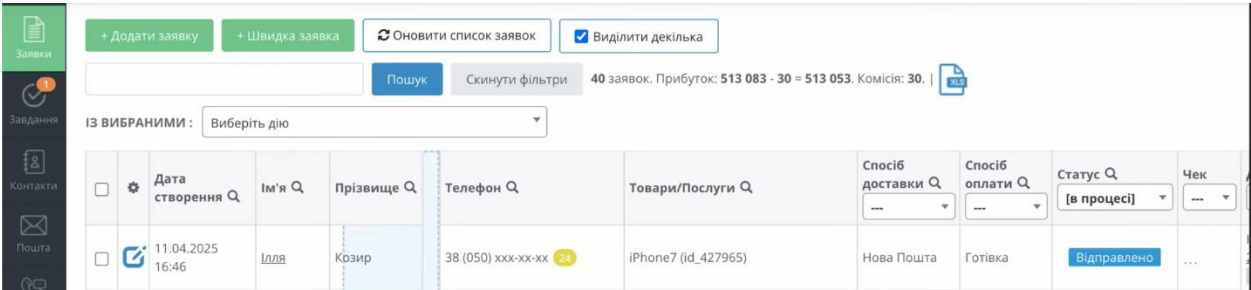


Рисунок 1 – Головне вікно програми

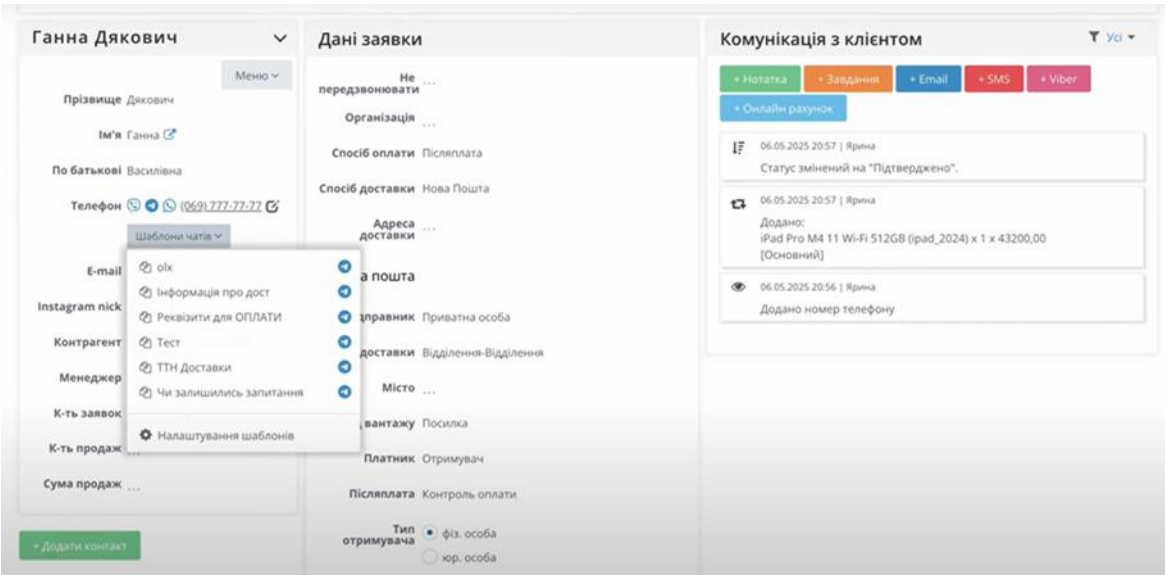


Рисунок 2 – Вікно комунікації з клієнтом

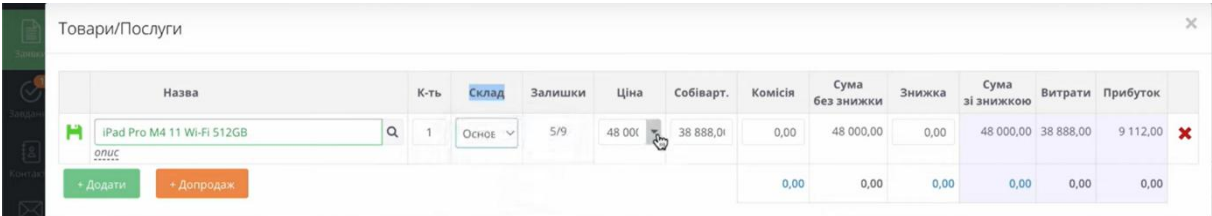


Рисунок 3 –Головне вікно продажу товарів

ДОДАТОК В – ТЕХНІЧНИЙ ОПИС

1 Призначення системи

Розроблена інформаційна система призначена для АРМ менеджера магазину мобільного зв'язку та формування звіту про продані товари, дозволяє редагувати та додавати дані і виконувати пошук по загальній базі даних товарів.

2 Умови застосування

Програма експлуатується на персональному комп'ютері (ПК) типа IBM PC/AT. Для роботи в діалоговому режимі використовується екран дисплея, клавіатура та мишу типу «миша». Для підтримки графічного режиму необхідний адаптер SVGA. Вхідні дані зберігаються на твердому диску комп'ютера. Програма працює під управлінням операційної системи Windows XP.

3 Опис задачі

Дана система покращує та спрощує роботу менеджера з продажу.

Задачі системи:

1. Додавання товарної позиції.
2. Редагування товарної позиції.
3. Внесення додаткової інформації про товар.
4. Внесення інформації про продаж товарів.
5. Пошук в загальній базі товарів.
6. Формування звіту про продані товари.

Виконання усіх перелічених задач реалізовано за допомогою окремих відповідних модулів розроблених за допомогою мови програмування Java.

Запуск програми здійснюється при відкритті файлу Main.EXE.

Вхідні та вихідні дані:

Кожен з модулів має особисті вхідні та вихідні дані, які перелічено нижче:

Модуль “Настройки підключення до БД ”

Вхідні дані: IP адрес, ім'я БД, логін, пароль.

Вихідні дані: користувач отримує середовище для виконання роботи, завантажене за логіном та паролем.

Модуль “Продажа товарів”

Вхідні дані: вибір виду роботи з переліку

Вихідні дані: користувач отримує середовище для виконання обраної роботи.

Модуль „Редагування товару ”

Вхідні дані: при редагуванні товару ми має змогу змінити будь яку необхідну інформацію.

ДОДАТОК Г – SQL СКРИПТ ДЛЯ РОЗРОБЛЕНОЇ БД

```
CREATE TABLE BatteryType
(
    idBatteryType INTEGER NOT NULL,
    batteryType VARCHAR(20) NOT NULL,
    PRIMARY KEY (idBatteryType));

CREATE TABLE BodyType
(
    idBodyType INTEGER NOT NULL,
    bodyType VARCHAR(20) NOT NULL,
    PRIMARY KEY (idBodyType));

CREATE TABLE Fillial
(
    idFillial INTEGER NOT NULL,
    `name` VARCHAR(40) NOT NULL,
    address VARCHAR(100) NOT NULL,
    PRIMARY KEY (idFillial));

CREATE TABLE Mobilephone
(
    idMobilephone INTEGER NOT NULL,
    price INTEGER NOT NULL,
    count INTEGER NOT NULL,
    idProduct INTEGER NOT NULL,
    idFillial INTEGER NOT NULL,
    PRIMARY KEY (idMobilephone));

CREATE TABLE MobilephoneParameters
(
    value VARCHAR(100) NOT NULL,
    idMobilephone INTEGER NOT NULL,
    idParameters INTEGER NOT NULL,
    PRIMARY KEY (idMobilephone,idParameters));

CREATE TABLE Model
```

```

(      idModel  INTEGER NOT NULL,
model  VARCHAR(35) NOT NULL,
PRIMARY KEY (idModel));

CREATE TABLE `Order`
(      idOrder  INTEGER NOT NULL,
`count`  INTEGER NOT NULL,
orderDate  DATE NOT NULL,
price  INTEGER NOT NULL,
idMobilephone  INTEGER NOT NULL,
PRIMARY KEY (idOrder));

CREATE TABLE `Parameters`
(      idParameters  INTEGER NOT NULL,
`parameter`  VARCHAR(45) NOT NULL,
PRIMARY KEY (idParameters));

CREATE TABLE PhoneType
(      idPhoneType  INTEGER NOT NULL,
phoneType  VARCHAR(20) NOT NULL,
PRIMARY KEY (idPhoneType));

CREATE TABLE Product
(      mark  VARCHAR(30) NOT NULL,
idProduct  INTEGER NOT NULL,
idBatteryType  INTEGER NOT NULL,
idBodyType  INTEGER NOT NULL,
idModel  INTEGER NOT NULL,
idPhoneType  INTEGER NOT NULL,
PRIMARY KEY (idProduct));

ALTER TABLE Mobilephone

```



```

ADD FOREIGN KEY R_17 (idProduct) REFERENCES Product(idProduct);

ALTER TABLE Mobilephone
ADD FOREIGN KEY R_9 (idFillial) REFERENCES Fillial(idFillial);

ALTER TABLE MobilephoneParameters
ADD FOREIGN KEY R_19 (idMobilephone) REFERENCES
Mobilephone(idMobilephone);

ALTER TABLE MobilephoneParameters
ADD FOREIGN KEY R_20 (idParameters) REFERENCES
Parameters(idParameters);

ALTER TABLE `Order`
ADD FOREIGN KEY R_18 (idMobilephone) REFERENCES
Mobilephone(idMobilephone);

ALTER TABLE Product
ADD FOREIGN KEY R_13 (idPhoneType) REFERENCES
PhoneType(idPhoneType);

ALTER TABLE Product
ADD FOREIGN KEY R_14 (idBatteryType) REFERENCES
BatteryType(idBatteryType);

ALTER TABLE Product
ADD FOREIGN KEY R_15 (idBodyType) REFERENCES
BodyType(idBodyType);

ALTER TABLE Product
ADD FOREIGN KEY R_16 (idModel) REFERENCES Model(idModel);

```

ДОДАТОК Д – Текст програми

```
package < database >
```

Листинг 1 - файл <Connector>

```
package database;
```

```
import java.sql.Connection;
```

```
import java.sql.ResultSet;
```

```
import java.sql.SQLException;
```

```
//Интерфейс коннектора.
```

```
public interface Connector {
```

```
    public Connection getConnection();
```

```
    public ResultSet executeQuery(String sql) throws  
SQLException;
```

```
    public boolean execute(String sql) throws SQLException;
```

```
    public void executeUpdate(String sql) throws SQLException;
```

```
    public void reconnect();
```

```
}
```

Листинг 2 - файл <MySQLConnector>

```
package database;
```

```
import gui.MessageDialog;
```

```
import java.sql.*;
```

```
import java.util.Properties;
```

```
import settings.Settings;
```

```
//Подключение к БД.
```

```
public class MySQLConnector implements Connector {
```

```
    private static MySQLConnector instance;
```

```
    private Connection connection;
```

```
    private MySQLConnector() {
```

```

    }
    public static Connector getInstance() {
        if (instance == null) {
            instance = new MySQLConnector();
        }
        return instance;
    }
    @Override
    public void reConnect() {
        try {
            if (connection != null) {
                connection.close();
            }
        } catch (SQLException ex) {
            new MessageDialog(null,
true).showErrorMessage("Неудалось закрыть соединение");
        }
        connect();
    }
    @Override
    public Connection getConnection() {
        if (connection == null) {
            connect();
        }
        return connection;
    }
    //Получить настройки подключения.
    private Properties getProperties() {
        Settings settings = Settings.getInstance();
        return settings.getProperties();
    }

```

//Подключение к БД. Возвращает true - если успешно, false - если не удалось.

```
private boolean connect() {
    try {
        Properties properties = getProperties();
        String ip = properties.getProperty("ip");
        String port = properties.getProperty("port");
        String dbname = properties.getProperty("dbname");
        String user = properties.getProperty("user");
        String password =
properties.getProperty("password");
        String url = "jdbc:mysql://" + ip + ":" + port +
"/" + dbname;
        connection = DriverManager.getConnection(url, user,
password);

        return true;
    } catch (SQLException ex) {
        return false;
    }
}

//SQL запрос с возвратом значений.
@Override
public ResultSet executeQuery(String sql) throws
SQLException {
    ResultSet resultSet = null;
    if (connect()) {
        Statement query =
connection.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
ResultSet.CONCUR_UPDATABLE);
        resultSet = query.executeQuery(sql);
    }
    return resultSet;
}
```

```

    }

    //SQL запрос с возвратом true - если успешно, иначе false.
    @Override
    public boolean execute(String sql) throws SQLException {
        boolean executed = false;
        if (connect()) {
            Statement query = connection.createStatement();
            executed = query.execute(sql);
        }
        return executed;
    }
    @Override
    public void executeUpdate(String sql) throws SQLException {
        if (connect()) {
            Statement query = connection.createStatement();
            query.executeUpdate(sql);
        }
    }
    @Override
    protected void finalize() throws Throwable {
        super.finalize();
        if (connection != null) {
            connection.close();
        }
    }
}

```

Листинг 3 - файл <Executor>

```

package database;

import gui.MessageDialog;
import java.sql.Connection;

```

```

import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
//Класс отвечающий за запросы к БД.
public class Executor {
    Connector connector;

    public Executor() {
        connector = MySQLConnector.getInstance();
    }
    //Поиск всех телефонов из БД. all = true показывает и
нелефоны которых нет в наличии
    public ResultSet findAllMobilephones(boolean all) {
        ResultSet resultSet = null;
        try {
            if (all) {
                resultSet =
connector.executeQuery(Queries.SELECT_ALL_FROM_MOBILEPHONES + "
ORDER BY `idMobilephone`");
            } else {
                resultSet =
connector.executeQuery(Queries.SELECT_ALL_FROM_MOBILEPHONES + " AND
m.`count` > 0 ORDER BY `idMobilephone`");
            }
        } catch (SQLException ex) {
            new MessageDialog(null,
true).showErrorMessage("Неудалось найти все телефоны.");
        }
        return resultSet;
    }
    //Поиск одного телефона по его ИД
    public ResultSet findMobilephoneByID(int idMobilephone) {

```

```

        ResultSet resultSet = null;
        Connection connection = connector.getConnection();
        try {
            PreparedStatement statement =
connection.prepareStatement(Queries.SELECT_MOBILEPHONE_BY_ID);
            statement.setInt(1, idMobilephone);
            resultSet = statement.executeQuery();
        } catch (SQLException ex) {
            new MessageDialog(null,
true).showErrorMessage("Неудалось найти телефон.");
        }
        return resultSet;
    }

    //Параметры определенного телефона
    public ResultSet mobileParametersById(int idMobilephone) {
        ResultSet resultSet = null;
        Connection connection = connector.getConnection();
        try {
            PreparedStatement statement =
connection.prepareStatement(Queries.SELECT_FROM_PARAMETER_BY_ID);
            statement.setInt(1, idMobilephone);
            resultSet = statement.executeQuery();
        } catch (SQLException ex) {
            new MessageDialog(null,
true).showErrorMessage("Неудалось найти параметры телефона.");
        }
        return resultSet;
    }

    //Все проданные телефоны. Таблица Ордерс
    public ResultSet findAllSelledPhones() {
        ResultSet resultSet = null;

```

```

        try {
            resultSet =
connector.executeQuery(Queries.SELECT_FROM_SELLED_MOBILEPHONES);
        } catch (SQLException ex) {
            new MessageDialog(null,
true).showErrorMessage("Список продаж пуст.");
        }
        return resultSet;
    }

    //Продажа телефона.
    public void buyMobilephone(int id, int idMobilephone, int
count, int price, int max) {
        try {
            Connection connection = connector.getConnection();
            PreparedStatement statement =
connection.prepareStatement(Queries.INSERT_INTO_ORDERS);
            statement.setInt(1, id);
            statement.setInt(2, count);
            statement.setDate(3, new
java.sql.Date(System.currentTimeMillis()));
            statement.setInt(4, price);
            statement.setInt(5, idMobilephone);
            statement.executeUpdate();
            takeMobilePhone(idMobilephone, count, max);
        } catch (SQLException ex) {
            new MessageDialog(null,
true).showErrorMessage("Неудалось заполнить таблицу покупок.");
        }
    }

    //Вычитание телефона из БД

```



```

        public void takeMobilePhone(int idMobilephone, int count,
int max) {
            try {
                Connection connection = connector.getConnection();
                PreparedStatement statement =
connection.prepareStatement(Queries.SELL_MOBILEPHONE);
                statement.setInt(1, max - count);
                statement.setInt(2, idMobilephone);
                statement.executeUpdate();
            } catch (SQLException ex) {
                new MessageDialog(null,
true).showErrorMessage("Неудалось изменить таблицу телефонов.");
            }
        }
        //Получить всободный индекс таблицы.
        public int getFreeId(String SQL) {
            int id = 1;
            try {
                ResultSet rs = connector.executeQuery(SQL);
                while (rs.next()) {
                    if (rs.getInt(1) == id) {
                        id++;
                    }
                }
                return id;
            } catch (SQLException ex) {
                return 0;
            }
        }
        //Получить любую таблицу справочник.
        public ResultSet getAllFromTable(String SQL) {
            ResultSet resultSet = null;

```

```

        try {
            resultSet = connector.executeQuery(SQL);
        } catch (SQLException ex) {
            new MessageDialog(null,
true).showErrorMessage("Неудалось получить мануал.");
        }
        return resultSet;
    }
    //Получить все модели телефонов.
    public ResultSet getAllModel() {
        return getAllFromTable(Queries.SELECT_ALL_FROM_MODEL);
    }
    //Получить все типы телефонов.
    public ResultSet getAllPhoneType() {
        return
getAllFromTable(Queries.SELECT_ALL_FROM_PHONETYPE);
    }
    //Получить все типы батарей.
    public ResultSet getAllBatteryType() {
        return
getAllFromTable(Queries.SELECT_ALL_FROM_BATTERYTYPE);
    }
    //Получить все типы корпуса.
    public ResultSet getAllBodyType() {
        return
getAllFromTable(Queries.SELECT_ALL_FROM_BODYTYPE);
    }

    //Получить все филиалы.
    public ResultSet getAllFillials() {
        return
getAllFromTable(Queries.SELECT_ALL_FROM_FILLIALS);
    }

```

```

    }
    //Получить все мобильные телефоны с подолнительным SKUЛ
КОДОМ.

    public ResultSet findAllMobilephones(StringBuilder SQL) {
        ResultSet resultSet = null;
        try {
            resultSet =
connector.executeQuery(Queries.SELECT_ALL_FROM_MOBILEPHONES + SQL);
        } catch (SQLException ex) {
            new MessageDialog(null,
true).showErrorMessage("Неудалось получить весь список телефонов.");
        }
        return resultSet;
    }
    //Добавить запись в справочник.
    public void insertManuals(String addSQL, int index, String
item) {
        try {
            Connection connection = connector.getConnection();
            PreparedStatement statement =
connection.prepareStatement(addSQL);
            statement.setInt(1, index);
            statement.setString(2, item);
            statement.executeUpdate();
        } catch (SQLException ex) {
            new MessageDialog(null,
true).showErrorMessage("Неудалось добавить '" + item + "'");
        }
    }
    //Обновить запись в справочнике.
    public void updateManuals(String addSQL, int index, String
item) {

```

```

        try {
            Connection connection = connector.getConnection();
            PreparedStatement statement =
connection.prepareStatement(addSQL);
            statement.setString(1, item);
            statement.setInt(2, index);
            statement.executeUpdate();
        } catch (SQLException ex) {
            new MessageDialog(null,
true).showErrorMessage("Неудалось добавить '" + item + "'");
        }
    }
    //Добавить продукт.
    public int addProduct(String mark, int batteryID, int
bodyID, int modelID, int phoneID) {
        int id = -1;
        try {
            Connection connection = connector.getConnection();
            PreparedStatement statement =
connection.prepareStatement(Queries.INSERT_INTO_PRODUCT);
            statement.setString(1, mark);
            id = getFreeId(Queries.INDEX_FROM_PRODUCT);
            statement.setInt(2, id);
            statement.setInt(3, batteryID);
            statement.setInt(4, bodyID);
            statement.setInt(5, modelID);
            statement.setInt(6, phoneID);
            statement.executeUpdate();
        } catch (SQLException ex) {
            new MessageDialog(null,
true).showErrorMessage("Неудалось добавить новый телефон.");
        }
    }

```

```

        return id;
    }

    //Добавить мобильный телефон.
    public void addMobilephone(int idMobilephone, int price,
int count, int idProduct, int idFillial) {
        try {
            Connection connection = connector.getConnection();
            PreparedStatement statement =
connection.prepareStatement(Queries.INSERT_INTO_MOBILEPHONES);
            statement.setInt(1, idMobilephone);
            statement.setInt(2, price);
            statement.setInt(3, count);
            statement.setInt(4, idProduct);
            statement.setInt(5, idFillial);
            statement.executeUpdate();
        } catch (SQLException ex) {
            new MessageDialog(null,
true).showErrorMessage("Неудалось добавить новый телефон.");
        }
    }

    //обновить продукт.
    public void updateProduct(int idProduct, String mark, int
batteryID, int bodyID, int modelID, int phoneID) {
        try {
            Connection connection = connector.getConnection();
            PreparedStatement statement =
connection.prepareStatement(Queries.UPDATE_PRODUCT);
            statement.setString(1, mark);
            statement.setInt(2, batteryID);
            statement.setInt(3, bodyID);
            statement.setInt(4, modelID);
            statement.setInt(5, phoneID);

```

```

        statement.setInt(6, idProduct);
        statement.executeUpdate();
    } catch (SQLException ex) {
        new MessageDialog(null,
true).showErrorMessage("Неудалось изменить телефон.");
    }
}

//Обновить мобильный телефон.
public void updateMobilephone(int idMobilephone, int price,
int count, int idFillial) {
    try {
        Connection connection = connector.getConnection();
        PreparedStatement statement =
connection.prepareStatement(Queryes.UPDATE_MOBILEPHONES);
        statement.setInt(1, price);
        statement.setInt(2, count);
        statement.setInt(3, idFillial);
        statement.setInt(4, idMobilephone);
        statement.executeUpdate();
    } catch (SQLException ex) {
        new MessageDialog(null,
true).showErrorMessage("Неудалось изменить телефон.");
    }
}

//Удалить мобильный телефон по ИД
public void removeMobilephone(Object id) {
    try {
        Connection connection = connector.getConnection();
        int idMobilephone = Integer.valueOf(id.toString());
        PreparedStatement statement =
connection.prepareStatement(Queryes.SELECT_PRODUCT_ID);
        statement.setInt(1, idMobilephone);

```

```

        ResultSet rs = statement.executeQuery();
        int idProduct = 0;
        if (rs.next()) {
            idProduct = rs.getInt(1);
        }
        statement =
connection.prepareStatement(Queries.DELETE_PARAMETERS_BY_ID);
        statement.setInt(1, idMobilephone);
        statement.execute();
        statement =
connection.prepareStatement(Queries.DELETE_MOBILEPHONE_BY_ID);
        statement.setInt(1, idMobilephone);
        statement.executeUpdate();
        try {
            removeProductByIdMobilephone(connection,
idProduct);
        } catch (SQLException ex) {
        }
    } catch (SQLException ex) {
        new MessageDialog(null,
true).showErrorMessage("Неудалось удалить телефон.");
    }
}

//Удалить продукт по ИД мобильного телефона.
private void removeProductByIdMobilephone(int
idMobilephone, Connection connection) throws SQLException {
    PreparedStatement statement;
    statement =
connection.prepareStatement(Queries.SELECT_PRODUCT_ID);
    statement.setInt(1, idMobilephone);
    ResultSet rs = statement.executeQuery();
    int idProduct = 0;

```

```

        if (rs.next()) {
            idProduct = rs.getInt(1);
        }
        statement =
connection.prepareStatement(Queries.DELETE_PRODUCT_BY_ID);
        statement.setInt(1, idProduct);
        statement.executeUpdate();
    }
    //Удалить продукт по ИД.
    private void removeProductByIdMobilephone(Connection
connection, int idProduct) throws SQLException {
        PreparedStatement statement =
connection.prepareStatement(Queries.DELETE_PRODUCT_BY_ID);
        statement.setInt(1, idProduct);
        statement.executeUpdate();
    }
    //Удалить параметр мобильного телефона
    public void removeMobilephoneParameter(int idMobilephone,
int idParameter) {
        try {
            Connection connection = connector.getConnection();
            PreparedStatement statement =
connection.prepareStatement(Queries.DELETE_PARAMETER_BY_IDM_IDP);
            statement.setInt(1, idMobilephone);
            statement.setInt(2, idParameter);
            statement.execute();
        } catch (SQLException ex) {
            new MessageDialog(null,
true).showErrorMessage("Неудалось удалить параметр телефона.");
        }
    }
    //Добавить параметр.

```



```

        public void insertParameter(int index, String param) {
            try {
                Connection connection = connector.getConnection();
                PreparedStatement statement =
connection.prepareStatement(Queries.INSERT_INTO_PARAMETERS);
                statement.setInt(1, index);
                statement.setString(2, param);
                statement.execute();
            } catch (SQLException ex) {
                new MessageDialog(null,
true).showErrorMessage("Неудалось добавить параметр.");
            }
        }

        //Добавить параметр мобильного телефона
        public void insertMobilephoneParameter(int idMobilephone,
int index, String text) {
            try {
                Connection connection = connector.getConnection();
                PreparedStatement statement =
connection.prepareStatement(Queries.INSERT_INTO_MOBILEPHONE_PARAMETE
RS);

                statement.setString(1, text);
                statement.setInt(2, idMobilephone);
                statement.setInt(3, index);
                statement.execute();
            } catch (SQLException ex) {
                new MessageDialog(null,
true).showErrorMessage("Неудалось добавить параметр телефона.");
            }
        }

        //Обновить параметр мобильного телефона.

```

```

        public void updateMobilephoneParameter(int idMobilephone,
int index, String text) {
            try {
                Connection connection = connector.getConnection();
                PreparedStatement statement =
connection.prepareStatement(Queries.UPDATE_MOBILEPHONE_PARAMETER);
                statement.setString(1, text);
                statement.setInt(2, idMobilephone);
                statement.setInt(3, index);
                statement.executeUpdate();
            } catch (SQLException ex) {
                new MessageDialog(null,
true).showErrorMessage("Неудалось добавить параметр телефона.");
            }
        }
        //Удалить продажу.
        public void removeOrder(int idOrder) {
            Connection connection = connector.getConnection();
            try {
                PreparedStatement statement =
connection.prepareStatement(Queries.DELETE_ORDER_BY_ID);
                statement.setInt(1, idOrder);
                statement.execute();
                try {
                    removeProductByIdMobilephone(idOrder,
connection);
                } catch (SQLException ex) {
                }
            } catch (SQLException ex) {
                new MessageDialog(null,
true).showErrorMessage("Неудалось удалить продажу.");
            }
        }

```

```

    }
    //Получить параметр по ИД
    public String getParamNameByID(int idParameter) {
        try {
            Connection connection = connector.getConnection();
            PreparedStatement statement =
connection.prepareStatement(Queries.SELECT_PARAMETER_NAME_BY_ID);
            statement.setInt(1, idParameter);
            ResultSet rs = statement.executeQuery();
            if (rs.next()) {
                return rs.getString(1);
            }
        } catch (SQLException ex) {
            new MessageDialog(null,
true).showErrorMessage("Неудалось найти ID телефона.");
        }
        return "";
    }

    //Удалить запись из справочника.
    public void removeManual(String sql, Integer index) {
        try {
            Connection connection = connector.getConnection();
            PreparedStatement statement =
connection.prepareStatement(sql);
            statement.setInt(1, index);
            statement.executeUpdate();
        } catch (SQLException ex) {
            new MessageDialog(null,
true).showErrorMessage("Неудалось найти ID телефона.");
        }
    }
}

```

```

        public void insertFillial(int index, String name, String
address) {
            try {
                Connection connection = connector.getConnection();
                PreparedStatement statement =
connection.prepareStatement(Queries.INSERT_INTO_FILLIALS);
                statement.setInt(1, index);
                statement.setString(2, name);
                statement.setString(3, address);
                statement.execute();
            } catch (SQLException ex) {
                new MessageDialog(null,
true).showErrorMessage("Неудалось добавить филиал.");
            }
        }

        public void updateFillial(int index, String name, String
address) {
            try {
                Connection connection = connector.getConnection();
                PreparedStatement statement =
connection.prepareStatement(Queries.UPDATE_FILLIAL);
                statement.setString(1, name);
                statement.setString(2, address);
                statement.setInt(3, index);
                statement.executeUpdate();
            } catch (SQLException ex) {
                new MessageDialog(null,
true).showErrorMessage("Неудалось изменить филиал.");
            }
        }

        public void removeFillial(int index) {

```

```

        try {
            Connection connection = connector.getConnection();
            PreparedStatement statement =
connection.prepareStatement(Queries.DELETE_FILLIAL_BY_ID);
            statement.setInt(1, index);
            statement.executeUpdate();
        } catch (SQLException ex) {
            new MessageDialog(null,
true).showErrorMessage("Неудалось удалить филиал.");
        }
    }
}

```

Листинг 4 - файл <Queries>

```

package database;

public interface Queries {

    public static String SELECT_ALL_FROM_MOBILEPHONES = "SELECT
m.`idMobilephone` AS '№', mo.model AS 'Модель', "
        + " p.mark AS 'Марка', pt.`phoneType` AS 'Тип
телефона', bdt.`bodyType` AS 'Тип корпуса', "
        + " bt.`batteryType` AS 'Тип батареи', m.price AS
'Цена', m.`count` AS 'Количество', f.`name` AS 'Отделение'"
        + " FROM mobilephone m, product p, batterytype bt,
model mo, phonetype pt, bodytype bdt, fillial f "
        + "WHERE m.`idProduct` = p.`idProduct` AND
p.`idPhoneType` = pt.`idPhoneType` AND p.`idModel` = mo.`idModel`"
        + " AND p.`idBodyType` = bdt.`idBodyType` AND
p.`idBatteryType` = bt.`idBatteryType` "
        + "AND m.`idFillial` = f.`idFillial`";

    public static String SELECT_FROM_MOBILEPHONES_BY_MODEL =
"SELECT m.`idMobilephone` AS '№', mo.model AS 'Модель', "

```

```

        + " p.mark AS 'Марка', pt.`phoneType` AS 'Тип
телефона', bdt.`bodyType` AS 'Тип корпуса', bt.`batteryType` "
        + "AS 'Тип батареи', m.price AS 'Цена', m.`count`
AS 'Количество', f.`name` AS 'Отделение' "
        + "FROM mobilephone m, product p, batterytype bt,
model mo, phonetype pt, bodytype bdt, fillial f "
        + "WHERE m.`idProduct` = p.`idProduct` AND
p.`idPhoneType` = pt.`idPhoneType` AND p.`idModel` = mo.`idModel`"
        + " AND mo.model = ? AND p.`idBodyType` =
bdt.`idBodyType` AND p.`idBatteryType` = bt.`idBatteryType` "
        + "AND m.`idFillial` = f.`idFillial`";

    public static String SELECT_FROM_SELLED_MOBILEPHONES =
"SELECT o.`idOrder` AS '№ заказа', m.`idMobilephone` "
        + "AS '№ телефона', mo.model AS 'Модель', p.mark AS
'Марка', o.price AS 'Цена', o.`count` "
        + "AS 'Количество', f.`name` AS 'Отделение',
o.`orderDate` AS 'Дата' FROM `order` o, mobilephone m,"
        + " product p, model mo, fillial f WHERE
o.`idMobilephone` = m.`idMobilephone`, m.`idFillial` =
f.`idFillial`"
        + " AND m.`idProduct` = p.`idProduct` AND
p.`idModel` = mo.`idModel` ORDER BY o.`idOrder`";

    public static String SELECT_FROM_PARAMETER_BY_ID = "SELECT
mp.`idParameters` AS '№', p.`parameter` AS 'Параметр', mp.`value` AS
'Значение' "
        + "FROM mobilephoneparameters mp, `parameters` p
WHERE mp.`idMobilephone` = ? "
        + "AND mp.`idParameters` = p.`idParameters` ORDER
BY mp.`idParameters`";

    public static String SELECT_MOBILEPHONE_BY_ID = "SELECT
p.`idModel`, p.mark, p.`idBodyType`, p.`idPhoneType`,

```

```

p.`idBatteryType`, m.price, m.`count`, m.`idProduct`, m.`idFillial`
"

        + "FROM mobilephone m, product p WHERE
m.`idMobilephone` = ? AND p.`idProduct` = m.`idProduct`";

        public static String SELECT_ALL_FROM_MODEL = "select
m.`idModel` AS '№', m.model AS 'Модель' from model m";

        public static String SELECT_ALL_FROM_BODYTYPE = "select
bt.`idBodyType` AS '№', bt.`bodyType` AS 'Тип корпуса' from bodytype
bt";

        public static String SELECT_ALL_FROM_BATTERYTYPE = "select
b.`idBatteryType` AS '№', b.`batteryType` AS 'Тип батареи' from
batterytype b";

        public static String SELECT_ALL_FROM_PHONETYPE = "select
pt.`idPhoneType` AS '№', pt.`phoneType` AS 'Тип телефона' from
phonetype pt";

        public static String SELECT_ALL_FROM_PARAMETERS_NAME =
"select p.`idParameters` AS '№', p.`parameter` AS 'Параметр' from
`parameters` p";

        public static String SELECT_ALL_FROM_FILLIALS = "SELECT
f.`idFillial` AS '№', f.`name` AS 'Название', f.address AS 'Адрес'
FROM fillial f";

        public static String SELECT_PRODUCT_ID = "SELECT
`idProduct` FROM mobilephone WHERE `idMobilephone` = ?";

        public static String SELECT_PARAMETER_NAME_BY_ID = "SELECT
p.`parameter` FROM `parameters` p WHERE p.`idParameters` = ?";

        public static String SELL_MOBILEPHONE = "UPDATE mobilephone
m SET m.`count` = ? WHERE m.`idMobilephone`=?"

        public static String INSERT_INTO_ORDERS = "INSERT INTO
`order` (`idOrder`, `count`, `orderDate`, price, `idMobilephone`)
VALUES(?, ?, ?, ?, ?)";

```

```

        public static String INSERT_INTO_MODEL = "INSERT INTO model
(`idModel`, model) VALUES (?, ?)";

        public static String INSERT_INTO_BODYTYPE = "INSERT INTO
bodytype (`idBodyType`, `bodyType`) VALUES (?, ?)";

        public static String INSERT_INTO_BATTERYTYPE = "INSERT INTO
batterytype (`idBatteryType`, `batteryType`) VALUES (?, ?)";

        public static String INSERT_INTO_PHONETYPE = "INSERT INTO
phonetype (`idPhoneType`, `phoneType`) VALUES (?, ?)";

        public static String INSERT_INTO_FILLIALS = "INSERT INTO
fillial(`idFillial`, `name`, address) VALUES(?, ?, ?)";

        public static String INSERT_INTO_PRODUCT = "INSERT INTO
product (mark, `idProduct`,
`idBatteryType`, `idBodyType`, `idModel`, `idPhoneType`) VALUES
(?,?,?,?,?,?)";

        public static String INSERT_INTO_MOBILEPHONES = "INSERT
INTO mobilephone(`idMobilephone`, price, `count`, `idProduct`,
`idFillial`)VALUES(?,?,?,?,?)";

        public static String INSERT_INTO_PARAMETERS = "INSERT INTO
`parameters` (`idParameters`, `parameter`) VALUES (?, ?)";

        public static String INSERT_INTO_MOBILEPHONE_PARAMETERS =
"INSERT INTO mobilephoneparameters (`value`, `idMobilephone`,
`idParameters`) VALUES (?, ?, ?)";


        public static String INDEX_FROM_PRODUCT = "SELECT
`idProduct` FROM product";

        public static String INDEX_INDEX_MOBILEPHONES = "SELECT
`idMobilephone` FROM mobilephone";


        public static String DELETE_PHONE_PARAMETERS = "DELETE FROM
mobilephoneparameters WHERE mobilephoneparameters.`idMobilephone` =
?";

```



```

        public static String DELETE_MOBILEPHONE = "DELETE FROM
mobilephone WHERE mobilephone.`idMobilephone` = ?";

        public static String DELETE_PARAMETERS_BY_ID = "DELETE FROM
mobilephoneparameters WHERE `idMobilephone` = ?";

        public static String DELETE_MOBILEPHONE_BY_ID = "DELETE
FROM mobilephone WHERE `idMobilephone` = ?";

        public static String DELETE_PRODUCT_BY_ID = "DELETE FROM
product WHERE `idProduct` = ?";

        public static String DELETE_PARAMETER_BY_IDM_IDP = "DELETE
FROM mobilephoneparameters WHERE `idMobilephone` = ? AND
`idParameters` = ?";

        public static String DELETE_ORDER_BY_ID = "DELETE FROM
`order` WHERE `idOrder` = ?";

        public static String DELETE_BODYTYPE_BY_ID = "DELETE FROM
bodytype WHERE `idBodyType` = ?";

        public static String DELETE_PHONETYPE_BY_ID = "DELETE FROM
phonetype WHERE `idPhoneType` = ?";

        public static String DELETE_BATTERYTYPE_BY_ID = "DELETE
FROM batterytype WHERE `idBatteryType` = ?";

        public static String DELETE_MODEL_BY_ID = "DELETE FROM
model WHERE `idModel` = ?";

        public static String DELETE_PARAMETER_BY_ID = "DELETE FROM
`parameters` WHERE `idParameters` = ?";

        public static String DELETE_FILLIAL_BY_ID = "DELETE FROM
fillial WHERE fillial.`idFillial` = ?";


        public static String UPDATE_MOBILEPHONE_PARAMETER = "UPDATE
mobilephoneparameters SET `value` = ? WHERE `idMobilephone` = ? AND
`idParameters` = ?";

        public static String UPDATE_PRODUCT = "UPDATE product SET
mark = ?, `idBatteryType` = ?, `idBodyType` = ?, `idModel` = ?,
`idPhoneType` = ? "

```

```

        + "WHERE `idProduct` = ?";

        public static String UPDATE_MOBILEPHONES = "UPDATE
mobilephone SET price = ?, `count` = ?, `idFillial` = ? WHERE
`idMobilephone` = ?";

        public static String UPDATE_BODYTYPE = "UPDATE bodytype SET
`bodyType` = ? WHERE `idBodyType` = ?";

        public static String UPDATE_PHONETYPE = "UPDATE phonetype
SET `phoneType` = ? WHERE `idPhoneType` = ?";

        public static String UPDATE_BATTERYTYPE = "UPDATE
batterytype SET `batteryType` = ? WHERE `idBatteryType` = ?";

        public static String UPDATE_MODEL = "UPDATE model SET model
= ? WHERE `idModel` = ?";

        public static String UPDATE_PARAMETERS = "UPDATE
`parameters` SET `parameter` = ? WHERE `idParameters` = ?";

        public static String UPDATE_FILLIAL = "UPDATE fillial SET
`name` = ? , address = ? WHERE `idFillial` = ?";
    }

```

Пакет <guilogic>

Листинг 1 - файл <AddFillialLogic>

```

package guilogic;

import gui.AddFillialDialog;

public class AddFillialLogic {

    private AddFillialDialog dialog;
    private boolean confirm;

    public AddFillialLogic() {
        dialog = new AddFillialDialog(null, true, this);
    }
}

```

```

public void showAddFillialDialog() {
    dialog.getjTextFieldName().setText("");
    dialog.getjTextAreaAddress().setText("");
    confirm = false;
    dialog.setVisible(true);
}

public void showEditFillialDialog(String name, String
address) {
    dialog.getjTextFieldName().setText(name);
    dialog.getjTextAreaAddress().setText(address);
    confirm = false;
    dialog.setVisible(true);
}

public String getName() {
    return dialog.getjTextFieldName().getText();
}

public String getAddress() {
    return dialog.getjTextAreaAddress().getText();
}

public boolean check() {
    if ("".equals(dialog.getjTextFieldName().getText())) {
        return false;
    }
    if ("".equals(dialog.getjTextAreaAddress().getText()))
{
        return false;
    }
    return true;
}

public boolean isConfirmed() {
    return confirm;
}

```

```

    public void confirm() {
        confirm = true;
        dialog.setVisible(false);
    }
    public void cancel() {
        confirm = false;
        dialog.setVisible(false);
    }
}

```

Листинг 2 - файл <BuyPhoneLogic>

```

package guillogic;

import database.Executor;
import gui.BuyPhoneDialog;

public class BuyPhoneLogic {

    private BuyPhoneDialog dialog;
    private Executor executor = new Executor();
    private int price = 0, max, value = 1, id;
    private boolean isSelled = false;

    public BuyPhoneLogic() {
        dialog = new BuyPhoneDialog(null, true, this);
    }

    public boolean isSelled() {
        return isSelled;
    }

    public void showDialog(int id, String mark, String model,
int price, int maximum) {

```

```

        this.price = price;
        this.id = id;
        this.max = maximum;
        dialog.initSpinner(maximum);
        dialog.setTextInLabel1("Марка: " + mark);
        dialog.setTextInLabel5("Модель: " + model);
        dialog.setTextInLabel7("Цена: " + price);
        isSelled = false;
        dialog.setVisible(true);
    }

    public void calculatePrice() {
        value = (int) dialog.getSpinner().getValue();
        dialog.setTextInLabel7("Цена: " + (price * value));
    }

    public void exitButton() {
        dialog.setVisible(false);
    }

    public void confirmButton() {
        int orderId = executor.getFreeId("SELECT * FROM
`order`");
        executor.buyMobilephone(orderId, id, value, value *
price, max);
        isSelled = true;
        dialog.setVisible(false);
    }
}

```

Листинг 3 - файл <FindPhoneLogic>

```
package guillogic;
```

```

import database.Executor;
import gui.FindPhoneDialog;
import java.sql.ResultSet;

```

```

import java.sql.SQLException;
import java.util.Scanner;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.DefaultComboBoxModel;

public class FindPhoneLogic {

    private FindPhoneDialog dialog;
    private final MobilephonesDialogLogic logic;

    public FindPhoneLogic(MobilephonesDialogLogic logic) {
        dialog = new FindPhoneDialog(null, true, this);
        this.logic = logic;
    }

    public void show() {
        Executor executor = new Executor();

        dialog.setjComboBox1(createComboBoxModel(executor.getAllModel()));

        dialog.setjComboBox2(createComboBoxModel(executor.getAllPhoneType()));
    };

    dialog.setjComboBox3(createComboBoxModel(executor.getAllBodyType()));
    ;

    dialog.setjComboBoxFillial(createComboBoxModel(executor.getAllFillials()));

        dialog.setAllCheckBoxUnselected();
        dialog.clearJTextFields();
        dialog.setVisible(true);
    }
}

```

```

        public void confirmButton() {
            logic.updateTable(getPhonesByParams());
            dialog.setVisible(false);
        }

        private ResultSet getPhonesByParams() {
            StringBuilder SQL = new StringBuilder(" AND m.`count` >
0 ");

            if (dialog.getjCheckBox4()) {
                int filllialId = new
Scanner(dialog.getjComboBoxFilllial().getSelectedItem().toString()).n
extInt();

                SQL.append("AND
m.`idFilllial`=").append(filllialId).append(" ");
            }

            if (dialog.getjCheckBox1()) {
                int modelId = new
Scanner(dialog.getjComboBox1().getSelectedItem().toString()).nextInt
();

                SQL.append("AND
mo.`idModel`=").append(modelId).append(" ");
            }

            if (dialog.getjCheckBox2()) {
                int phoneTypeId = new
Scanner(dialog.getjComboBox2().getSelectedItem().toString()).nextInt
();

                SQL.append("AND
pt.`idPhoneType`=").append(phoneTypeId).append(" ");
            }

            if (dialog.getjCheckBox3()) {
                int bodyTypeId = new
Scanner(dialog.getjComboBox3().getSelectedItem().toString()).nextInt
();

```

```

        SQL.append("AND
bdt.`idBodyType`=").append(bodyTypeId).append(" ");
    }
    if (dialog.getjCheckBox5()) {
        int start = dialog.getTextField1();
        int end = dialog.getTextField2();
        if (start > end || end == 0) {
            end = Integer.MAX_VALUE;
        }
        SQL.append("AND m.price >=").append(start).append("
AND m.price <=").append(end).append(" ");
    }
    return new Executor().findAllMobilephones(SQL);
}

private DefaultComboBoxModel<String>
createComboBoxModel(ResultSet rs) {
    DefaultComboBoxModel<String> boxModel = new
DefaultComboBoxModel();
    try {
        while (rs.next()) {
            boxModel.addElement(rs.getInt(1) + " " +
rs.getString(2));
        }
    } catch (SQLException ex) {

Logger.getLogger(FindPhoneLogic.class.getName()).log(Level.SEVERE,
null, ex);

    }
    return boxModel;
}
}

```

Листинг 4 - файл <SettingsDialogLogic>


```

package guilogic;

import com.sun.xml.internal.fastinfoset.util.CharArray;
import database.MySQLConnector;
import gui.MessageDialog;
import gui.SettingsDialog;
import java.awt.Frame;
import java.util.Properties;
import settings.Settings;

public class SettingsDialogLogic {

    private Settings settings;
    private Properties properties;
    private SettingsDialog settingsDialog;

    public SettingsDialogLogic(Frame parent) {
        settingsDialog = new SettingsDialog(parent, true,
this);
    }

    public void showPropertiesDialog() {
        initTextFields();
        settingsDialog.setVisible(true);
    }

    private void initTextFields() {
        settings = Settings.getInstance();
        properties = settings.getProperties();

settingsDialog.setTextInField1(properties.getProperty("ip"));

settingsDialog.setTextInField2(properties.getProperty("port"));

```

```

settingsDialog.setTextInField4(properties.getProperty("dbname"));

settingsDialog.setTextInField3(properties.getProperty("user"));

settingsDialog.setPassword(properties.getProperty("password"));
    }
    private boolean checkFields() {
        if ("".equals(settingsDialog.getTextFromField1())) {
            return false;
        }
        if ("".equals(settingsDialog.getTextFromField2())) {
            return false;
        }
        if ("".equals(settingsDialog.getTextFromField4())) {
            return false;
        }
        if ("".equals(settingsDialog.getTextFromField3())) {
            return false;
        }
        return true;
    }
    public void confirm() {
        if (checkFields()) {
            properties.setProperty("ip",
settingsDialog.getTextFromField1());
            properties.setProperty("port",
settingsDialog.getTextFromField2());
            properties.setProperty("dbname",
settingsDialog.getTextFromField4());
            properties.setProperty("user",
settingsDialog.getTextFromField3());

```

```

        char[] password = settingsDialog.getPassword();
        StringBuilder sb = new StringBuilder(new
CharArray(password, 0, password.length, false));
        properties.setProperty("password", sb.toString());
        settings.saveProperties();
        MySQLConnector.getInstance().reConnect();
        settingsDialog.setVisible(false);
    } else {
        new MessageDialog(null,
true).showErrorMessage("Поля не должны быть пустыми.");
    }
}

public void cancel() {
    settingsDialog.setVisible(false);
}
}

```

Листинг 5 - файл <AddManualLogic>

```

package guillogic;

import gui.AddManualDialog;

public class AddManualLogic {

    private AddManualDialog addManualDialog;
    private String aValue;
    private boolean confirm;

    AddManualLogic() {
        this.addManualDialog = new AddManualDialog(null, true,
this);
    }
}

```

```

    }
    public void showAddDialog(String name) {
        addManualDialog.getjLabel2().setText(name);
        aValue = "";
        addManualDialog.getjTextField1().setText(aValue);
        confirm = false;
        addManualDialog.setVisible(true);
    }
    public void showEditDialog(String name, String aValue) {
        addManualDialog.getjLabel2().setText(name);
        addManualDialog.getjTextField1().setText(aValue);
        confirm = false;
        addManualDialog.setVisible(true);
    }
    public String getValue() {
        return addManualDialog.getjTextField1().getText();
    }
    public boolean cheak() {
        if
("".equals(addManualDialog.getjTextField1().getText())) {
            return false;
        }
        return true;
    }
    public boolean isConfirmed() {
        return confirm;
    }
    public void confirm() {
        confirm = true;
        addManualDialog.setVisible(false);
    }
    public void cancel() {

```

```

        confirm = false;
        addManualDialog.setVisible(false);
    }
}

```

Листинг 6 - файл <AddParametersLogic>

```

package guillogic;

import database.Executor;
import database.Queries;
import gui.AddParametersDialog;
import gui.MessageDialog;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.Scanner;
import javax.swing.DefaultComboBoxModel;
import javax.swing.JComboBox;

public class AddParametersLogic {

    private AddParametersDialog dialog;
    private Executor executor = new Executor();
    private DefaultComboBoxModel model;
    private ResultSet rs;
    private int idMobilephone, index;
    private String aValue;
    private boolean editing = false;

    public AddParametersLogic() {
        dialog = new AddParametersDialog(null, true, this);
    }

    public void initComboBox() {

```

```

        rs =
executor.getAllFromTable(Queries.SELECT_ALL_FROM_PARAMETERS_NAME);
        model = new DefaultComboBoxModel();
        try {
            while (rs.next()) {
                model.addElement(rs.getObject(1) + " " +
rs.getObject(2));
            }
        } catch (SQLException ex) {
            new MessageDialog(null,
true).showErrorMessage("Неудалось инициализировать комбобокс.");
        }
        dialog.getjComboBoxParametr().setEnabled(true);
        updateJComboBox(model);
    }
    public void initComboBox(int idParameter) {
        model = new DefaultComboBoxModel();
        model.addElement(idParameter + " " +
executor.getParamNameByID(idParameter));
        updateJComboBox(model);
        dialog.getjComboBoxParametr().setEnabled(false);
    }
    void updateJComboBox(DefaultComboBoxModel model) {
        dialog.getjComboBoxParametr().setModel(model);
    }
    private void addNewParameterName(String param) {
        index =
executor.getFreeId(Queries.SELECT_ALL_FROM_PARAMETERS_NAME);
        executor.insertParameter(index, param);
        model.addElement(index + " " + param);
        updateJComboBox(model);
    }
}

```

```

public void confirm() {
    JComboBox jComboBox = dialog.getjComboBoxParametr();
    Object returned, selectedItem;
    returned = selectedItem = jComboBox.getSelectedItem();
    if (jComboBox.getSelectedIndex() == -1) {
        returned = cheack(selectedItem);
        if (Boolean.valueOf(returned.toString())) {
            addNewParameterName(selectedItem.toString());
            addParameter(index);
        }
    } else {
        if (!editing) {
            addParameter(Integer.valueOf(returned.toString().split(" ")[0]));
        } else {
            updateParameter(Integer.valueOf(returned.toString().split(" ")[0]));
        }
    }
    dialog.setVisible(false);
}

public void cancel() {
    dialog.setVisible(false);
}

private void addParameter(int index) {
    if (!"".equals(dialog.getjTextFieldValue().getText()))
    {
        executor.insertMobilephoneParameter(idMobilephone,
index, dialog.getjTextFieldValue().getText());
    } else {
        new MessageDialog(null, true).showErrorMessage("Все
поля обязательны для заполнения.");
    }
}

```

```

        }
    }

    private void updateParameter(int index) {
        if (!"".equals(dialog.getjTextFieldValue().getText()))
    {
        executor.updateMobilephoneParameter(idMobilephone,
index, dialog.getjTextFieldValue().getText());
        } else {
            new MessageDialog(null, true).showErrorMessage("Все
поля обязательны для заполнения.");
        }
    }

    public void showAddDialog(int idMobilephone) {
        this.idMobilephone = idMobilephone;
        editing = false;
        initComboBox();
        clearFields();
        dialog.setVisible(true);
    }

    public void showEditDialog(int idMobilephone, int
idParameter, String aValue) {
        this.idMobilephone = idMobilephone;
        this.aValue = aValue;
        this.editing = true;
        initComboBox(idParameter);
        initField();
        dialog.setVisible(true);
    }

    private Object cheack(Object selectedItem) {
        for (int i = 0; i < model.getSize() - 1; i++) {

```



```

        if
(name(model.getElementAt(i)).equalsIgnoreCase(selectedItem.toString(
))) {

            return model.getElementAt(i);

        }

    }

    return "true";

}

private String name(Object selectedItem) {
    Scanner scan = new Scanner(selectedItem.toString());
    scan.next();
    return scan.nextLine().trim();
}

private void clearFields() {
    dialog.getjComboBoxParametr().setSelectedIndex(-1);
    dialog.getjTextFieldValue().setText("");
}

private void initField() {
    dialog.getjTextFieldValue().setText(aValue);
}

}

```

Листинг 7 - файл <MobilephonesDialogLogic>

```

package guilogic;

import database.Executor;
import gui.MessageDialog;
import gui.MobilephoneDialog;
import java.awt.Frame;
import java.sql.ResultSet;
import javax.swing.JTable;
import model.TableModel;

```

```

public class MobilephonesDialogLogic {

    private MobilephoneDialog mobilephonesDialog;
    private BuyPhoneLogic buyPhoneLogic = new BuyPhoneLogic();
    private FindPhoneLogic findPhoneLogic = new
FindPhoneLogic(this);
    private Executor executor = new Executor();
    ResultSet empty = executor.mobileParametersById(0);
    private boolean solded = true;

    public MobilephonesDialogLogic(Frame parent) {
        mobilephonesDialog = new MobilephoneDialog(parent,
true, this);
    }
    public boolean isSelled() {
        return solded;
    }
    public void findButton() {
        findPhoneLogic.show();
    }
    public void closeButton() {
        clearTableParameter();
        mobilephonesDialog.setVisible(false);
    }
    public void clearTableParameter() {
        mobilephonesDialog.getTable2().setModel(new
TableModel(empty));
    }
    public void buyButton() {
        JTable table = mobilephonesDialog.getTable1();
        int selectedRow = table.getSelectedRow();

```

```

        if (sold && selectedRow != -1) {
            int id = (int) table.getValueAt(selectedRow, 0);
            String model = (String)
table.getValueAt(selectedRow, 1);
            String mark = (String)
table.getValueAt(selectedRow, 2);
            int price = (int) table.getValueAt(selectedRow, 6);
            int count = (int) table.getValueAt(selectedRow, 7);
            buyPhoneLogic.showDialog(id, mark, model, price,
count);

            if (buyPhoneLogic.isSelled()) {
                showSelledPhones();
            }
        }

        public void updateTable(ResultSet rs) {
            mobilephonesDialog.getTable1().setModel(new
TableModel(rs));
            mobilephonesDialog.setVisible(true);
        }

        public void selected() {
            int column;
            JTable table = mobilephonesDialog.getTable1();
            int selectedRow = table.getSelectedRow();
            if (selectedRow != -1) {
                column = sold ? 0 : 1;
                ResultSet resultSet =
executor.mobileParametersById((int) table.getValueAt(selectedRow,
column));

                if (resultSet == null) {
                    new MessageDialog(null,
true).showErrorMessage("Неудалось считать данные из БД.");

```

```

        } else {
            mobilephonesDialog.getTable2().setModel(new
TableModel(resultSet));
        }
    }

    public void setVisible() {
        mobilephonesDialog.setVisible(true);
    }

    public void showPhones() {
        ResultSet resultSet =
executor.findAllMobilephones(false);
        if (resultSet != null) {
            mobilephonesDialog.getTable1().setModel(new
TableModel(resultSet));
        }
        solded = true;
    }

    public void showSelledPhones() {
        ResultSet resultSet = executor.findAllSelledPhones();
        if (resultSet != null) {
            mobilephonesDialog.getTable1().setModel(new
TableModel(resultSet));
        }
        solded = false;
    }
}

```

```
package <settings>
```

Листинг 1 - файл <settings>

```
package settings;
```

```

import gui.MessageDialog;
import java.io.File;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.util.Properties;
//Настройки подключения к БД.
//Класс синглтон, для того чтоб в программе не было нескольких
объектов этого класса.
public class Settings {
    private static Settings instance;
    private Properties properties = new Properties();
    //Приватный конструктор
    private Settings() {
        restoreProperties();
    }
    //Получить объект класса, если не существует то создается.
    public static Settings getInstance() {
        if (instance == null) {
            instance = new Settings();
        }
        return instance;
    }
    //Стандартные настройки подключения.
    public void useDeffaultSetting() {
        properties.setProperty("ip", "localhost");
        properties.setProperty("port", "3306");
        properties.setProperty("dbname", "master");
        properties.setProperty("user", "root");
        properties.setProperty("password", "root");
    }
    //Восстановить настройки из файла.

```

```

        private void restoreProperties() {
            try {
                properties.load(new FileReader(new
File("connection.properties")));
            } catch (IOException ex) {
                new MessageDialog(null,
true).showErrorMessage("Неудалось считать файл настроек.");
                useDeffaultSetting();
            }
        }
        //Сохранить настройки в файл.
        public void saveProperties() {
            try {
                properties.store(new FileWriter(new
File("connection.properties")), "Настройки подключения к БД.");
            } catch (IOException ex) {
                new MessageDialog(null,
true).showErrorMessage("Неудалось сохранить файл настроек.");
            }
        }
        public Properties getProperties() {
            return properties;
        }
    }
}

```