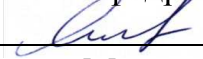


МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ КОРАБЛЕБУДУВАННЯ
ІМЕНІ АДМІРАЛА МАКАРОВА

Навчально-науковий інститут комп'ютерних наук та управління проектами
Кафедра інформаційних управляючих систем та технологій

"Допущений до захисту"

Завідувач кафедри ІУСТ



к.т.н, доц. Михелєв І.Л.

"__" "____" 2024 р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття ступеня вищої освіти "магістр"

(освітньо-кваліфікаційний рівень)

на тему **Дослідження та аналіз систем прогнозування обсягів
продажу на основі різних факторів на прикладі мережі магазинів
автозапчастин**

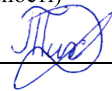
Виконав: студент 6 курсу, групи 6145м
напряму підготовки (спеціальності)

124 – "Системний аналіз".

(шифр і назва напряму підготовки, спеціальності)

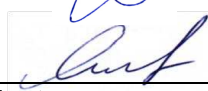
Тихомиров Д.А.

(прізвище та ініціали)



Керівник Михелєв І.Л.

(прізвище та ініціали)



м. Миколаїв – 2024 р.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ КОРАБЛЕБУДУВАННЯ
ІМЕНІ АДМІРАЛА МАКАРОВА

Інститут, факультет Навчально-науковий інститут комп'ютерних наук та управління проектами

Кафедра Інформаційних управляючих систем та технологій

Рівень вищої освіти другий (магістерський)

Спеціальність 124 – "Системний аналіз"

Освітня програма "Системний аналіз"

"ЗАТВЕРДЖУЮ"

Гарант освітньої програми

д.ф-м.н, проф. Ушкац М.В.

"___" _____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ
на здобуття ступеня вищої освіти "магістр"

Студенту Тихомиров Дмитро Андрійович

1. Тема проекту (роботи) Дослідження та аналіз систем прогнозування обсягів продажу на основі різних факторів на прикладі мережі магазинів автозапчастин

керівник проекту (роботи) Михелєв Ігор Леонідович,

затверджені наказом вищого навчального закладу від "08" 10 2024 року № 1055-уч.

2. Строк подання студентом проекту (роботи) 15 грудня 2024 року.

3. Вихідні дані до проекту (роботи) ДСТУ щодо обробки інформації, літературні джерела, технічна документація на існуючі аналоги систем, матеріали практики.

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

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

- Вступ.

- Аналіз предметної області і постановка задачі.

- Розробка проекту автоматизованої системи.

- Реалізація проекту автоматизованої системи (технічна документація по проекту).

- Розділ з охорони праці.
- Висновки.
- Список використаної літератури.
- Додатки: загальний опис підсистеми, керівництво користувача, керівництво адміністратора, програма і методика випробувань, графічні інтерфейси, код програми.

5. Перелік графічного матеріалу

- 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. Екранні форми.

6. Консультанти розділів проекту (роботи)

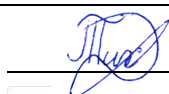
Розділ	Прізвище, ініціали та посада Консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1.	Михелєв І.Л.	05.09.2023	22.12.2023
2.	Михелєв І.Л.	23.12.2023	15.06.2024
3.	Михелєв І.Л.	15.06.2024	25.09.2024
4.	Михелєв І.Л.	25.09.2024	14.11.2024
5.	Гурець Н.В.	25.09.2024	14.11.2024
6.	Гурець Н.В.	25.09.2024	14.11.2024

7. Дата видачі завдання 05 вересня 2024 р.

КАЛЕНДАРНИЙ ПЛАН

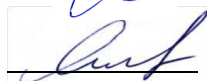
№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1.	Вивчення і опис предметної області	22.12.2024	
2.	Розробка концепції ІС	15.03.2024	
3.	Розробка проектних рішень по ІС	15.06.2024	
4.	Реалізація проекту, розробка робочої документації	25.09.2024	
5.	Вирішення питань з охорони праці	14.11.2024	
6.	Виконання графічної частини роботи	14.11.2024	
7.	Оформлення пояснювальної записки	14.11.2024	
8.	Подання роботи на кафедральний захист	16.11.2024	
9.	Подання роботи рецензенту	20.11.2024	
10.	Подання роботи на державний захист	15.12.2024	

Студент



Тихомиров Д.А.

Керівник проекту (роботи)



к.т.н., доцент, Михелєв І.Л.

Анотація

Дипломна робота присвячена дослідженню та вибору алгоритмів прогнозування даних для прийняття рішень адміністратором мережі магазинів автозапчастин для розробки підсистеми «Адміністратор магазинів»

При виконанні дипломної роботи: розроблено концепції інформаційної системи рішення з організаційного забезпечення, рішення з інформаційного забезпечення, рішення з технічного забезпечення, рішення з програмного забезпечення. Виконано кодування програмних модулів, виконано тестування отриманого програмного забезпечення, розроблено додатки.

Метою створення системи підвищення ефективності організації магазину шляхом автоматизації робочого процесу.

Робота викладена на 121 сторінках, містить 32 рисунка, 8 таблиць та 3 додатки.

Ключові слова: системний аналіз, системи прогнозування, магазини автозапчастин.

The annotation

Graduation is devoted to the study and selection of data prediction algorithms for decision-making by the administrator of the network of auto parts stores, for the development of the integration system "Administrator of the auto parts store"

At performance of degree work: concepts of information system of the decision on organizational maintenance, the decision on information maintenance, the decision on maintenance, the decision on the software are developed. Coding of program modules was completed, the received software was tested, applications were developed.

The purpose of the system is to increase the efficiency of the organization of the store by automating the workflow.

The work is presented in 121 pages, contains 32 figures, 8 tables and 3 annexes.

Keywords: systems analysis, forecasting systems, auto parts stores.

Зміст

Вступ	9
1 Дослідження алгоритмів прогнозування даних	11
1.1 Дослідження алгоритмів прогнозування даних для прийняття рішень	11
1.2 Алгоритм моделювання сезонності	12
1.3 Алгоритм мультиплікативних моделей часового ряду	17
2 Розробка концепції	27
2.1 Характеристика об'єкта	27
2.2 Аналіз предметної області	27
2.3 Схема організаційної структури управління об'єктом	29
2.3.1 Аналіз інформаційної структури управління об'єктом	31
2.3.2 Документообіг підприємства.	32
2.3.3 Розв'язувана проблема.	35
2.3.4 Вимоги до системи	35
2.4 Функціональна задача	36
2.4.1 Склад функціональних завдань системи	36
2.4.2 Вибір типових проектних рішень	37
2.4.3 Технічне забезпечення	39
2.5 Постановка задачі	40
3 Розробка технічного проекту підсистеми “адміністратор магазинів”	42
3.1.1 Діаграма варіантів використання	42
3.1.2 Загальносистемні проектні рішення. побудова функціональної моделі	43
3.2 Рішення з організаційного забезпечення.	46
3.3 Рішення з інформаційного забезпечення	47
3.3.1 Опис інформаційного забезпечення	47
3.3.3 Побудова фізичної моделі даних	48
3.4 Рішення з технічного забезпечення	53
3.4.1 Склад та структура технічного забезпечення системи	53
3.4.2 Опис комплексу технічних засобів	53
3.5 Рішення з програмного забезпечення	55
3.5.1 Розробка статичної моделі програмного забезпечення	55

3.5.2 Діаграма класів	55
3.5.3 Розробка проекту інтерфейсу	57
4 Розробка робочої документації	63
5 Охорона праці	65
5.1 Вступ	65
5.2 Аналіз небезпечних і шкідливих факторів в магазині	67
5.3 Розрахунок системи штучного освітлення приміщення обчислювальної техніки магазину.	70
6 Охорона навколишнього середовища	76
6.1 Вступ	76
6.2 Забруднення навколишнього середовища в магазині автозапчастин	76
6.3 Заходи щодо запобігання забруднень навколишнього середовища	78
7 Розрахунок економічного ефекту від впровадження підсистеми алгоритмів прогнозування даних	81
7.1 Вступна частина	81
7.2 Розрахунок вартості створення підсистеми магазину з продажу автозапчастин	82
7.3 Розрахунок економічної ефективності	85
7.4 Висновки з економічної ефективності	87
Висновки	88
Список використаних джерел	89
Додаток А	91
Додаток Б	96
Додаток В	102

Вступ

Тема даної дипломної роботи: дослідження алгоритмів прогнозування даних для прийняття рішень адміністратором мережі магазинів автозапчастин.

Актуальність даної теми обумовлена тим, що велика частина населення має автомобілі, котрі часто потребують дрібного ремонту з використанням автозапчастин. Але купувати автозапчастини будь-де, від невідомих виробників, не маючи жодних гарантій – це недозволена розкіш та й зайва небезпека. Тому покупці вирішують звернутись до сертифікованого магазину. З зростанням попиту з'являється необхідність в прогнозуванні закупівельного товару та його оборот в зв'язку з сезонністю

Прогнозування даних вирішить відразу кілька проблем: збільшить прибуток магазинів, зменшить витрати на складування зайвого товару, спростить аналітичний відділ.

Об'єктом дипломної роботи є магазин автозапчастин.

Суб'єктом дипломної роботи є програмне забезпечення автоматизації магазину.

Метою даної дипломної роботи є пошук та вибір алгоритму прогнозування даних та розробка підсистеми, яка буде спрощувати та вдосконалювати роботу магазину з продажу автозапчастин.

1. Дослідження алгоритмів прогнозування даних

1.1 Дослідження алгоритмів прогнозування даних для прийняття рішень

Дослідження алгоритмів прогнозування даних та розробка проекту підсистеми управління автомагазином здійснюється з метою вдосконалення певних процесів, які відбуваються під час поповнення складу магазину певними автозапчастинами.

Предмет дослідження – алгоритми прогнозування даних.

Об'єкт дослідження – процес наповнення складу затребуваними запчастинами.

Магазин займається продажем запасних частин для легкових автомобілів. Тому в складському господарстві завжди має бути достатня кількість товару, щоб його можна було замовити і направити в магазин. У той же час виробникам, у яких магазин купує продукцію, потрібно надавати довгострокові заявки, щоб вони могли включити дану інформацію у власні виробничі плани. До певного моменту замовлення постачальникам формувалися виходячи з даних про продажі в попередні періоди, помножених на експертні коефіцієнти зростання або зниження по окремих товарних групах. Такий підхід призводив до помилок в прогнозуванні запасів. Через істотне відхилення планованих значень від фактичних, оборотність товарів знижувалася, а підприємство зазнавало додаткові фінансові втрати у вигляді вартості зайвих закуплених деталей і неодержаного доходу від продажу незамовлених запасних частин, а також витрат на зберігання запасів. Щоб вирішити ці проблеми, було прийнято рішення розглянути та вибрати алгоритми прогнозування даних для закупівлі товару та своєчасного наповнення складу.

1.2 Алгоритм моделювання сезонності

У компанії була виявлена сезонність попиту на продавані запасні частини. Перший пік припадає на березень-квітень, в цей час спостерігається зростання ділової активності в регіоні, а другий відзначається у вересні-жовтні - в цей час корпоративні клієнти активно реалізують закупівельні бюджети. Таку яскраво виражену періодичність росту і зниження попиту в загальному випадку можна описати наступною гармонійної функцією:

$$f(k) = A_0 + A \sin\left(\frac{2\pi k}{T} + \varphi\right)$$

де A_0 – середньорічний попит в натуральних одиницях;

k - місяць розрахункового періоду;

T - період (тижнів);

φ - зсув по тимчасовій осі.

Для практичних обчислень цю формулу потрібно перетворити, щоб можна було включити шукані величини модельної функції у вигляді лінійних коефіцієнтів. Крім цього, варто мати на увазі і відмінність в амплітудах двох річних піків попиту. Після обліку зазначених факторів вихідна модельна функція приймає наступний вигляд:

$$f(k) = A_0 + A \sin\left(\frac{2\pi k}{T}\right) + B_1 \cos\left(\frac{2\pi k}{T}\right) + A_1 \sin\left(\frac{4\pi k}{T}\right) + B_1 \cos\left(\frac{4\pi k}{T}\right)$$

Тут коефіцієнти A , B_1 , A_2 і B_2 відображають амплітуди сезонних коливань з періодом півроку і рік. Таким чином, формула являє собою три перші члена розкладання в ряд Фур'є, який тут проявився в зв'язку із заявленою періодичністю. Вирази для розрахунку коефіцієнтів відомі з теорії Фур'є-аналізу. Вони можуть бути визначені за наступними формулами:

$$A_0 = \frac{1}{12} \sum_{k=1}^{12} g_k$$

$$A_1 = \frac{1}{12} \sum_{k=1}^{12} \left[2g_k \sin\left(\frac{2\pi k}{12}\right) \right]$$

$$B_1 = \frac{1}{12} \sum_{k=1}^{12} \left[2g_k \cos\left(\frac{2\pi k}{12}\right) \right]$$

$$A_2 = \frac{1}{12} \sum_{k=1}^{12} \left[2g_k \sin\left(\frac{4\pi k}{12}\right) \right]$$

$$B_2 = \frac{1}{12} \sum_{k=1}^{12} \left[2g_k \cos\left(\frac{4\pi k}{12}\right) \right]$$

Показник g_k в даному випадку відображає фактичні продажі за аналізований період. Проміжні коефіцієнти потрібно розрахувати для кожного значення за кожен місяць року. Варто зазначити, що вибір періодичності (в ситуації, що розглядається це два річних піку) не обмежується описуваної моделлю. Наприклад, якщо специфіка

Місяць	A0(продаж и 2017р.)	A 1	A 2	B 1	B 2	П рогноз продажу на 2018 р.
1	61	6 1	1 05,66	1 05,66	1	
2	44	7 6,21	7 6,21	4	44	
3	75	1 50	0	0	150	
4	120	2 07,85	- 207,85	- 120	120	
5	84	8 4	- 145,49	- 145,49	4	
6	67	0	0	- 134	34	
7	77	- 77	1 33,37	- 133,37	7	
8	59	- 102,19	1 02,19	- 59	59	
9	76	- 152	0	0	152	
10	80	- 138,56	- 138,56	8 0	80	
11	115	- 115	- 199,19	1 99,19	15	
12	50	0	1 00	1 00	00	
Коефі цієнти	75,67	- 0,47	- 22,81	- 5,25	- 2,83	

попиту на товари в натуральних показниках передбачає сезонність за порами року, то, додаючи в модельну функцію необхідні коефіцієнти з відповідною періодичністю, можна її розширити.

Таблиця 1.1 - Розрахунок прогнозу продажів в натуральних показниках (на прикладі номенклатурної одиниці «Омивач скла, White»)

Продемонструємо описану методику на прикладі розрахунку прогнозних обсягів продажів товару «Омивач скла, White» на 2018 рік. Так, в стовпці А (таблиця 1.1) зазначений порядковий номер місяця, в стовпці В - кількісні дані по продажах номенклатурної одиниці у відповідному місяці 2017 року.

Стовпці С-Е заповнюються математичними виразами, які відповідають творам під знаком суми в формулах, що використовуються для розрахунку коефіцієнтів модельної функції. Проміжне значення, наприклад, за перший місяць для кожного коефіцієнта визначається наступним чином:

$$A1_{січ} = 2 \cdot B3 \cdot \sin(2 \cdot \pi() \cdot A3/12) = 2 \cdot 61 \cdot \sin(2 \cdot \pi() \cdot 1/12) = 61$$

$$A2_{січ} = 2 \cdot B3 \cdot \sin(4 \cdot \pi() \cdot A3/12) = 2 \cdot 61 \cdot \sin(4 \cdot \pi() \cdot 1/12) = 105.66$$

$$B1_{січ} = 2 \cdot B3 \cdot \cos(2 \cdot \pi() \cdot A3/12) = 2 \cdot 61 \cdot \cos(2 \cdot \pi() \cdot 1/12) = 105.66$$

$$B2_{січ} = 2 \cdot B3 \cdot \sin(4 \cdot \pi() \cdot A3/12) = 2 \cdot 61 \cdot \sin(4 \cdot \pi() \cdot 1/12) = 61$$

Після того як знайдені всі проміжні значення для кожного місяця, визначаються шукані коефіцієнти А1, В1, А2 і В2. Потім отримані результати підставляються в модельну функцію і розраховується прогноз за обсягом попиту в натуральних показниках для кожного місяця (стовпець G). Наприклад, прогноз продажів по номенклатурної одиниці «Омивач скла, White» на вересень 2018 року становитиме:

$$B15 + C15 \cdot \sin(2 \cdot \pi() \cdot A11/12) + D15 \cdot \sin(4 \cdot \pi() \cdot A11/12) + E15 \cdot \cos(2 \cdot \pi() \cdot A11/12) + F15 \cdot \cos(4 \cdot \pi() \cdot A11/12) = 75.67 - 0.47 \cdot \sin(2 \cdot \pi() \cdot 9/12) - 22.81 \cdot \sin(4 \cdot \pi() \cdot 9/12) - 5.25 \cdot \cos(2 \cdot \pi() \cdot 9/12) - 2.83 \cdot \cos(4 \cdot \pi() \cdot 9/12) = 79$$

Розрахунок закупівельних цін

Ми визначили обсяг продажів номенклатурної одиниці на 2018 рік в

натуральних показників. У той же час для організації закупівель компанію цікавить прогноз попиту в закупівельних цінах. Так, у другому стовпці таблиці 2 наведені помісячні ціни, за якими запчастини розглянутого виду закуповувалися в 2017 році.

Таблиця 1.2 - Прогноз попиту номенклатурної одиниці «Омивач скла, White» на 2016 рік в закупівельних цінах

Місяць	іна в 2017 р. У.о.	П рогноз ціни на 2018 р. У.о.	Прогноз продаж на 2018 р. В натуральному виразу	пр огноз попиту, У.о.
Січень	2,97	1 6,84	50	84 2,2
Лютий	2,97	1 7,17	54	92 7,18
Березень	2,97	1 7,5	78	13 64,69
Квітень	2,97	1 7,82	99	17 64,38
Травень	5,39	1 8,15	98	17 78,5
Червень	5,39	1 8,47	78	14 40,97
Липень	5,39	1 8,8	59	11 09,2
Серпень	5,39	1 9,13	60	11 47,56
Вересень	5,54	1 9,45	79	15 36,71
Жовтень	5,54	1 9,78	95	18 78,91
Листопад	6,09	2 0,1	90	18 09,36

Грудень		2		13
	6,09	0,43	68	89,24

Якщо побудувати графік виходячи із зазначених цін і потім додати на нього лінійний тренд, то можна отримати рівняння такого вигляду:

$$y = 0.326x + 12.606$$

де y - це ціна закупівлі;

x - порядковий номер місяця.

Це означає, що в даному випадку ціна в середньому зростає на 0,326 у.о на місяць із значенням 12,606 у.о на початок періоду аналізу. Підставивши в рівняння вихідні дані за цінами, отримаємо помісячний прогноз по ним на 2018 рік. Потім ці значення множаться на раніше розрахований прогноз продажів в натуральних показниках, що дає прогноз попиту на товари по закупівельній вартості.

Прогнозування продаж

Описана модель впроваджується в підсистему «Адміністратор магазинів». При першому плануванні попиту формується документ «План продажів» на кожен з 12 календарних місяців, наступних за поточним. В кожному наступному місяці ця операція повторюється. У підсумку в документах «План продажів» за раніше запланованим періодів буде формуватися різниця між спочатку вказаною кількістю товарів і значенням, отриманим при використанні більш «свіжих» відомостей про фактичні продажі товарів. Таке ковшне планування дозволяє щомісяця актуалізувати дані.

Обмеження моделі

При формуванні заявок постачальникам додатково використовується фільтрація товарів, заснована на аналізі інтенсивності їх попередніх продажів (присутності товарів в документах реалізації). Товари також класифікуються за принципом стабільності величини продажів (XYZ-аналіз).

До класу X відносяться товари, за якими коефіцієнт варіації (відношення середньоквадратичного відхилення до середньоарифметичного значення продажів товару) знаходиться в діапазоні від 0 до 40 відсотків, до класу Y - в діапазоні понад 40 до 100 відсотків, до класу Z - понад 100 відсотків. Наприклад, за даними за 2017 рік через всієї проданої номенклатури близько 10 відсотків склали товари групи X , що характеризуються стабільною величиною продажів. Близько 30 відсотків довелося на

групу Y, з середньою «передбачуваністю».

Решта - погано прогнозована група Z. Здавалося б, моделювання схильне тільки 40 відсотків номенклатури (групи X і Y), але на ділі на них припадає понад 80 відсотків товарів в натуральному вираженні. При цьому замовлення продукції стабільного попиту можна повністю віддати на відкуп запропонованої моделі, оскільки отриманих таким шляхом даних цілком достатньо для побудови надійного прогнозу. А ось для інших категорій продукції (товари групи Y і Z) є потреба у додатковому аналізі і експертна оцінка.

Крім того, до обмежень моделі відноситься неможливість аналізу зовнішніх чинників, наприклад, поява нового великого конкурента або загальноекономічних ринкових коливань. Це пов'язано з тим, що модельні розрахунки проводяться на основі інформації минулих періодів, тому зростає потреба в якісній експертну грошову оцінку. Для підвищення її ефективності використовуються дані про середні короткострокових продажах товарів і їх оборотності.

1.3 Алгоритм мультиплікативних моделей часового ряду

Існують дві основні моделі часового ряду: адитивна і мультиплікативна.

Формула адитивної моделі:

$$Y_t = T_t + S_t + e_t$$

Формула мультиплікативної моделі:

$$Y_t = T_t \times S_t + e$$

t - час (місяць або інший період деталізації);

Y - значення величини;

T - тренд;

S - сезонні зміни;

e - шум.

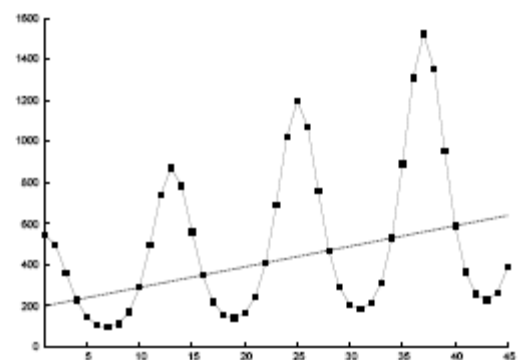
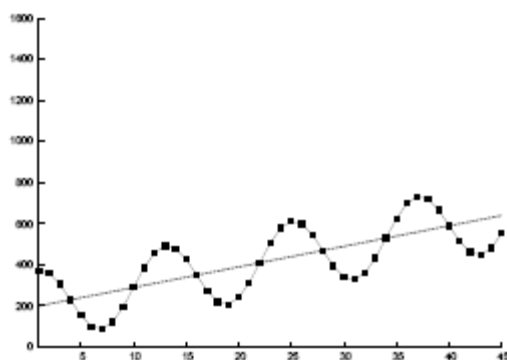


Рисунок 1.1 - Приклади рядів: зліва - по адитивної моделі; праворуч - по

мультиплікативної.

В даному прикладі буде розглядатись мультиплікативна модель.

Для будь-яких інших даних, можливо краще підійшла б адитивна модель. Дізнатися на практиці, яка модель підходить краще, можна або інтуїтивно, або методом проб і помилок.

Виділення тренда

У формулах моделей рядів динаміки ($Y_t = T_t + S_t + e_t$ і $Y_t = T_t S_t + e_t$) фігурує тренд T_t , такий тренд ми будемо називати «точним».

Найпростіший спосіб отримання наближеного тренда - згладжування ряду методом змінного середнього з періодом згладжування рівним максимальному періоду сезонних коливань. Згладжування майже повністю усуне сезонні коливання і шум.

В рядах з деталізацією по місяцях згладжування треба зробити по 12 точках (тобто по 12 місяців). Формула змінного середня з періодом згладжування 12 місяців:

Де M_t — значення змінного середнього в точці t ; Y_t — значення величини часового ряду в точці t .

Прологарифмуємо рівняння мультиплікативної моделі, і якщо шум e_t не дуже великий, то отримаємо аддитивну модель. $\ln\{Y_t\} = \ln\{T_t\} + \ln\{S_t\} + \varepsilon_{1,t}$

Тут $\varepsilon_{1,t}$ також позначає шум. Тренд виділимо (ковзаючим середнім за 12 місяців) саме для такої перетвореної моделі. На рисунку 3 - графіки і показника і тренда M_t .

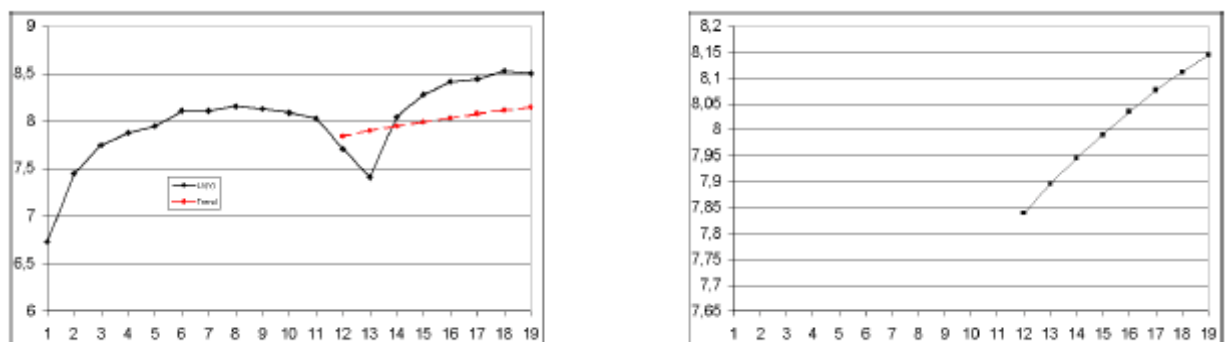


Рисунок 1.2 - Графік прологарифмованної величини показника і тренда M_t змінного середнього по 12-ти місяців. Зліва на одному графіку і величина і тренд. Праворуч - тренд в збільшеному масштабі. По осі X - номери періодів.

Прогноз трендів

Тренд отримали, тепер потрібно його спрогнозувати. Прогноз можна б було отримати, наприклад, методом експоненціального згладжування, але оскільки ми хочемо прогнозувати максимально простим методом, то зупинимося на звичайній параметричній апроксимації. Як функції наближення використовуємо наступний набір:

Лінійна функція: $y = a + b \times t$.

логарифмічна функція: $y = a + b \times \ln(t)$

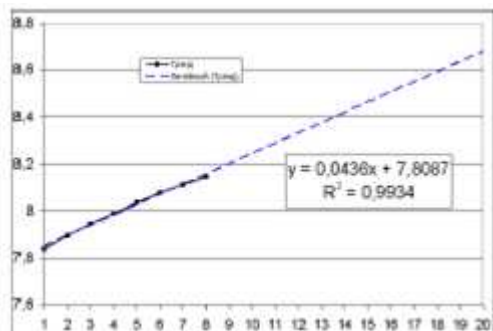
Поліном другого ступеня: $y = a + b \times t + c \times t^2$

Степенева функція: $y = a \times t^b$

Експоненціальна функція: $y = a \times e^{b \times t}$

Якість наближення оцінюється за величиною достовірності апроксимації R^2 . Чим ближче ця величина до 1 - тим краще функція наближає тренд.

На рисунках 1.3, 1.4, 1.5, 1.6 і 1.7 зробили апроксимацію тренда різними функціями і кожна функція апроксимації продовжена на 12 точок вперед. І ще одна



апроксимація - на рисунку 1.8, поліномом 5-тій мірі.

Рисунок 1.3 – Пряма, $y = a + bt$

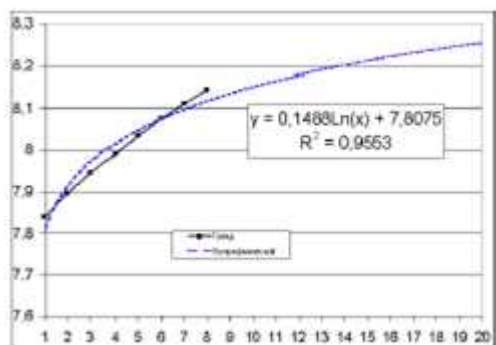


Рисунок 1.4 – Логарифмічна функція, $y = a + b \times \ln(t)$

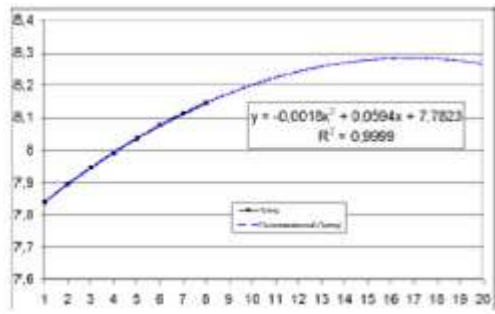


Рисунок 1.5 – Парабола, $y = a + bt + ct^2$

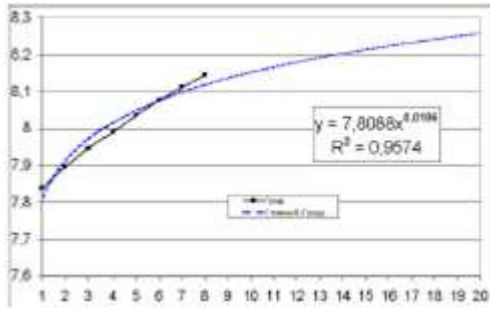


Рисунок 1.6 – Степенева функція: $y = a \times t^b$

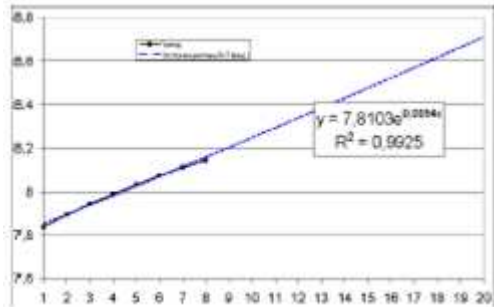


Рисунок 1.7 - Експоненціальна функція: $y = a \times e^{b \times t}$

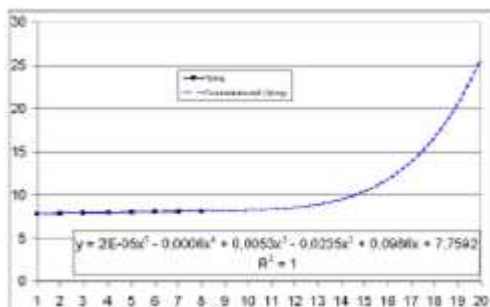


Рисунок 1.8 – Полином п'ятого ступеня.

За рисунками бачимо, що значення R^2 найближче до одиниці у параболи.

Наступна за якістю апроксимація - пряма лінія. Хоча формально парабола апроксимує краще за всіх, але її поведінка, особливо перевал у віддалених точках, видається не дуже правдоподібним. Тоді можна взяти апроксимацію прямий, але буде знайдений компроміс: середнє арифметичне між параболою і прямий.

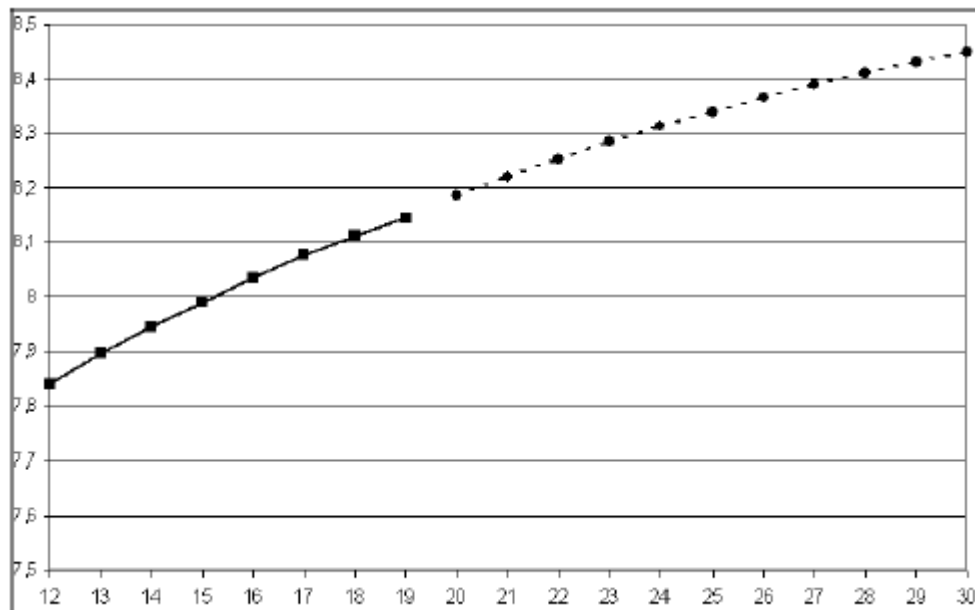


Рисунок 1.9- Тренд M_t та його прогноз. По осі X — номер періода.

Результат прогнозу тренда M_t — на рисунку 9.

Алгоритм прогнозу показника

Прогноз тренда у нас є. Тепер можна зробити прогноз самого показника.

Формула очевидна:

$$\ln(Y_{t+1}) = 12 \times M_{t+1} - \ln(Y_t) - \ln(Y_{t-1}) - \dots - \ln(Y_{t-10})$$

$$Y_{t+1} = \exp(\ln(Y_{t+1}))$$

До періоду $t = 19$ у нас є фактичні дані. Для $t = 20..31$ у нас є прогнозований тренд M_t , а значення показника ми будемо вважати послідовно, спочатку для $t = 20$, потім для $t = 21$ і т.д.

Результати прогнозу зображені на рисунку 10 і в таблиці 2.

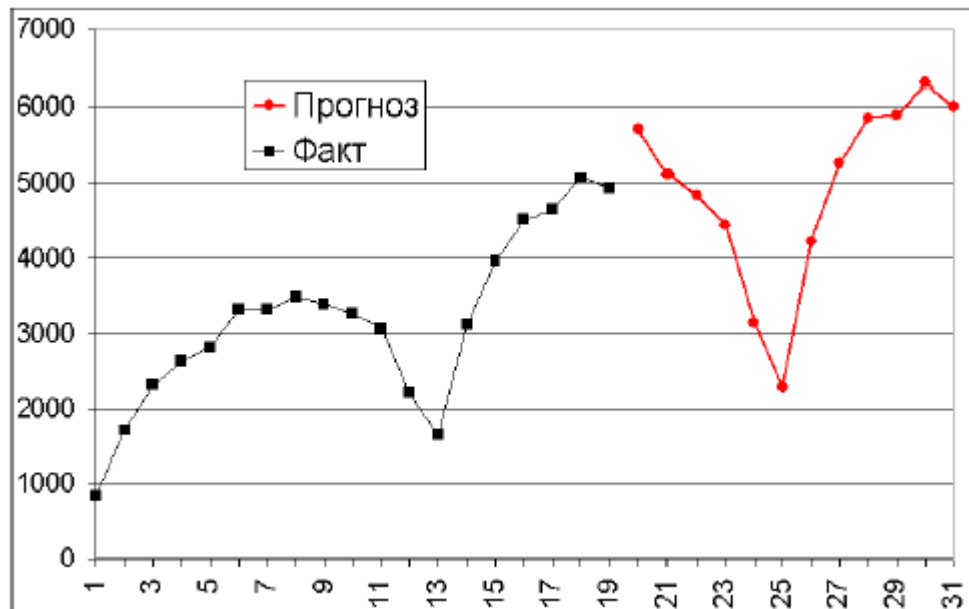


Рисунок 1.10 - Прогноз показника. По осі Х - номер періоду.

Порівняння прогнозу і реальних даних

На рисунку 11 - графіки прогнозу і фактичних даних.

У таблиці 1.3 наведено порівняння реальних даних і прогнозованих. Пораховані помилки прогнозу, абсолютні: Прогноз-Факт; і відносні: $100\% * (\text{Прогноз} - \text{Факт}) / \text{Факт}$.

Таблиця 1.3 Прогноз показників.

Період	№ Періода	M	Ln(Y)	Y
2017-2	20	8,1861	8,6494	5707
2017-3	21	8,2205	8,5408	5119
2017-4	22	8,2531	8,4816	4825
2017-5	23	8,2839	8,3987	4441
2017-6	24	8,3129	8,0533	3144
2017-7	25	8,3401	7,7367	2291
2017-8	26	8,3655	8,3488	4225
2017-9	27	8,3891	8,5675	5258
2017-10	28	8,4109	8,6765	5864
2017-11	29	8,4309	8,6833	5904
2018-12	30	8,4491	8,7487	6303
2018-1	31	8,4655	8,7007	6007

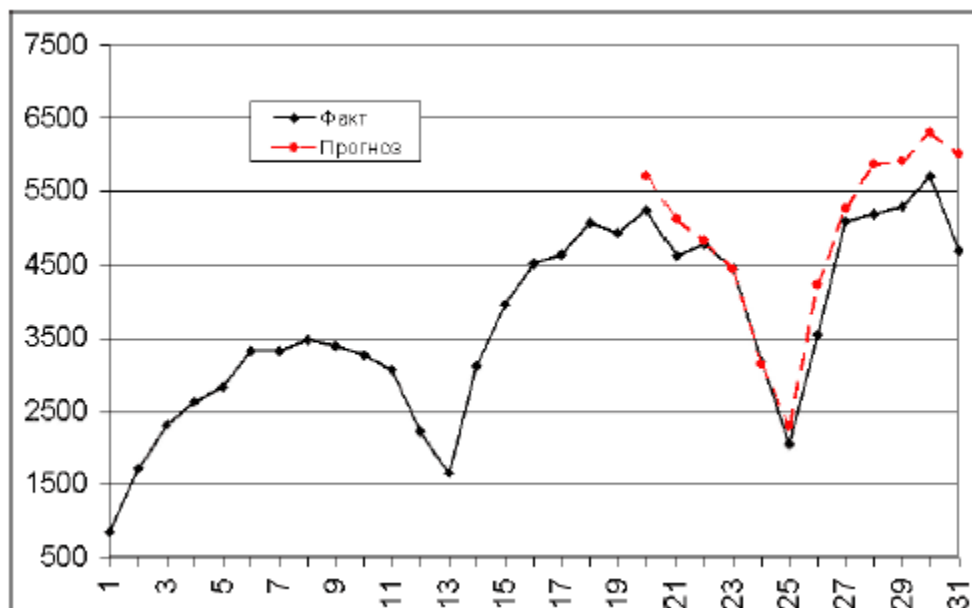


Рисунок 1.11 - Фактичні дані і спрогнозовані. По осі Х - номер періоду.

Навіть якщо модель дуже добре описує динаміку реальних даних, що в загальному велика рідкість, то залишаються ще шуми, які вносять свою помилку. Наприклад, якщо рівень шуму становить 10% від значення показника, то і помилка прогнозу буде не менше 10%. Плюс, як мінімум, ще кілька відсотків помилки додаються через невідповідність моделі і динаміки реальних даних.

Таблиця 1.4 - Порівняння фактичних і прогнозованих даних.

П еріод	№ Периода	акт	Пр огноз	Ош ибка, абс.	Ош ибка, %
20 17-2	20	233	57 07	474	9
20 17-3	21	625	51 19	494	11
20 17-4	22	776	48 25	49	1
20 17-5	23	457	44 41	-16	0
20 17-6	24	169	31 44	-25	-1
20 17-7	25	054	22 91	237	12
20 17-8	26	549	42 25	676	19
20 17-9	27	087	52 58	171	3
20 17-10	28	187	58 64	677	13

20 17-11	29	287	59 04	617	12
20 17-12	30	700	63 03	603	11
20 18-1	31	689	60 07	131 8	28

В розділах 1.2 та 1.3 описані та наведені приклади для алгоритму моделювання сезонності та алгоритму мультиплікативних моделей часового ряду. Виходячи з проведеного аналізу, був обраний алгоритм моделювання сезонності, так як він дає прогноз використовуючи данні за менший період часу, та дає більш точні вихідні данні.

Отже, використання алгоритму моделювання сезонності надає можливість автоматизувати процес, прогнозувати закупівлю для кожної групи товарів, уникнути помилок, які можуть виникнути при ручному аналізі великого обсягу даних та зекономити час при прийманні рішення щодо товару, який береться на реалізацію.

2 Розробка концепції

2.1 Характеристика об'єкта

– Діяльністю даного об'єкта управління є роздрібна торгівля автозапчастинами для легкових автомобілей імпортного та вітчизняного вироблення, зокрема для:

- - двигунів;
- - ходової частини;
- - трансмісії;
- - гальмівної системи;
- - системи зчеплення;
- - рульового керування;
- - приводу ГРМ;
- - приводу колес;
- - системи кондиціювання;
- - системи опалення та вентиляції;
- - системи палива;
- - системи запалювання;
- - випускних систем;
- - електроприладів автомобіля;
- - системи очистки вікон;
- - автомобільної косметики.

– 2.2 Аналіз предметної області

Основними функціями магазину з продажу автозапчастин є продаж товару та ведення бухгалтерії.

При прийманні товару, адміністратор оглядає стан товару та упаковки, в разі пошкодження або дефекту, товар відправляється, як брак, постачальнику. Процес замовлення та занесення товару на склад магазину складається з наступних етапів:

- Адміністратор складає накладну на товар та видає шляховий лист вантажникам та водіям .

- Приймання товару здійснюється за накладною виданою постачальником.
- В журнал обліку товарів заноситься інформація про всі атрибути товару та його вартість.

Процес продажу товару складається з наступних етапів:

- Продавець-консультант допомагає клієнту з вибором необхідних товарів та товарний чек з обраним товаром.
- Клієнт з товарним чеком виписаним продавцем-консультантом звертається до каси.
- Касир приймає товарний чек і після оплати товару видає покупцеві касовий чек та накладну на товар.
- Пред'явивши накладну на товар в складі магазину, вантажник видає товар покупцеві.
- В журналі обліку товару відмічається проданий товар і в разі відсутності достатньої кількості на складі, робиться помітка "замовити".

Накладна містить наступні дані:

- номер накладної;
- дата;
- від кого;
- кому;
- номер позиції в накладній;
- найменування товару;
- одиниці виміру товару;
- кількість;
- ціна;
- сума.

При оформленні звіту для адміністрації бухгалтер вказує наступні дані:

- найменування придбаного товару;
- кількість одиниць придбаного товару;
- загальна сума від продажу;

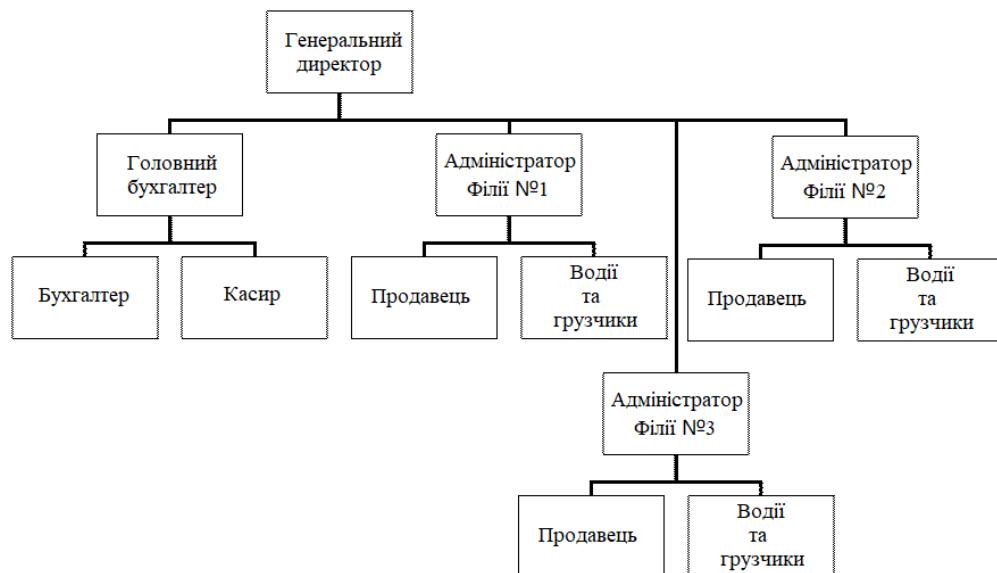
Даний спосіб ведення документації має безліч недоліків. По-перше, процес продажу займає багато часу. Оскільки продавець-консультант повинен дізнатись про наявність товару на складі, або виконати пошук товару в торговому залі. По-друге, кількість клієнтів магазину при належній популярності буде зростати, та продавець-консультант буде не в змозі обробити велику кількість запитів. Тому необхідна автоматизація роботи персоналу.

Підсистема необхідна, так як це дозволить уникнути наступних проблем:

- зменшити час на обробку запиту покупця;
- уникнути помилок при веденні документації;
- уникнути помилок при формуванні звітів;
- уникнути втрати інформації.

Основними вимогами до підсистеми «Адміністратор магазинів» є:

- Безпечне зберігання даних.
 - Коректне виконання функцій.
 - Зручність в використанні
- 2.3 Схема організаційної структури управління об'єктом
- На рисунку 2.1 представлена схема організаційної структури управління об'єктом



- Рисунок 2.1 – Схема організаційної структури управління об'єктом
- Директор(власник) магазину визначає напрямок діяльності, контроль за належним виконанням персоналом його обов'язків, контролює

фінансову діяльність магазину, визначає розмір заробітної плати та грошової винагороди.

- Менеджер або адміністратор виконує наступні функції:
- - визначає номенклатуру автозапчастин, яка на його думку повинна продаватися в магазині та задовольняти купівельний попит;
- - аналізує ринок виробників автозапчастин та якість виробленої ними продукції;
- - проводить аналіз оптових постачальників автозапчастин, тобто порівнює асортимент автозапчастин на складі постачальників, якість запропонованої ними продукції та відсутність контрафакту, рівень цін, систему знижок та репутацію самого постачальника;
- - поповнює асортимент товарів та слідкує за його наявністю на складі для задовільнення купівельного попиту;
- - формує актуальні ціни на товар відповідно до ринкових цін та отримання прибутку.

– Виходячи з проведеного аналізу адміністратор формує кошик закупівель для насичення складу. Процес закупівлі і доставки товару підтверджується чеком про оплату і паперовими накладними. Розміщення товару на складі проводиться хаотично. При обробці запиту покупця продавець витрачає значну кількість часу на пошук необхідного товару. Документом, що підтверджує факт здійснення продажу є видаткова накладна видана продавцем. Аналіз продаж за окремими видами товарів, за період часу або по конкретному покупцеві - утруднений, що зводить аналітичну діяльність до мінімуму не дозволяючи оперативно реагувати на запити покупців і кон'юнктури ринку.

- Продавець - здійснює реалізацію товару.

Водії та вантажники - займаються доставкою, розвантаженням і завантаженням товарів

Головний бухгалтер - контролює фінансову діяльність підприємства.

Бухгалтер - веде первинну бухгалтерську документацію (виписка рахунків, накладних, прибуткових - видаткових ордерів), займається операціями, пов'язаними з оплатою товарів безготівковим способом.

Касир - займається операціями, пов'язаними з оплатою товарів готівковим способом.

2.3.1 Аналіз інформаційної структури управління об'єктом

Запропонована концепція програмного забезпечення дозволить автоматизувати і впорядкувати роботу персоналу магазину.

Директор(власник) буде своєчасно, або по запиті, отримувати необхідні звіти, для прийняття рішень щодо управління магазином.

Менеджер або адміністратор має змогу удосконалити виконання своїх функцій за допомогою системи, що розроблюється.

Також дана система дозволить поєднати функції головного бухгалтера, бухгалтера і касира, а також спростить аналітичний відділ, знизив витрати на заробітну плату працівників.

Час на формування замовлення та його обробку значно скоротиться, що дозволить задовольняти потреби більшого числа клієнтів і відповідно покращить фінансові показники підприємства.

На рисунку 2.2 представлена структура інформаційних потоків підприємства підсистеми «Адміністратор магазинів».

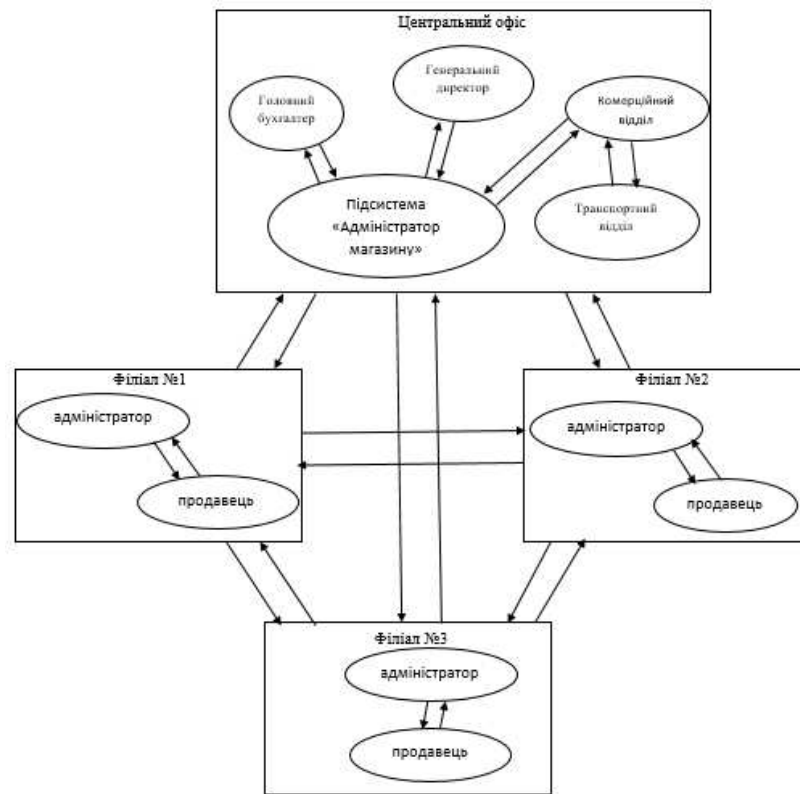


Рисунок 2.2 - Структура інформаційних потоків підприємства підсистеми «Адміністратор магазинів».

2.3.2 Документообіг підприємства.

Документообіг даного підприємства заснований на ручній виписці прибутково-видаткових документів, тобто після здійснення вибору товару, заповнюється товарний чек. Товарний чек є документом для оплати товару на касі. Після оплати товару на касі покупець отримує касовий чек, який підтверджує факт оплати обраного товару і два примірника товарних накладних для отримання товару. При отриманні товару на складі продавець-консультант і покупець ставлять підписи на обох примірниках товарної накладної. Один примірник товарної накладної покупець залишає собі, другий примірник товарної накладної залишається у продавця-консультанта для подальшої передачі в бухгалтерію з метою формування звіту. В кінці звітного періоду бухгалтер на підставі накладних, виданих за звітний період, формує звіт.

Для отримання магазином заздалегідь оплаченого товару у постачальників заповнюються бланки товарно-транспортних накладних, які видаються водіям і є підставою для отримання товару і його перевезення.

Бланки вищезазначених документів представлені на рисунках 2.3-2.4

ТОВАРНЫЙ ЧЕК № _____ от _____ 200__ г.

ИП (ООО) _____

ОГРН _____ от _____ выдано _____

Св-во о пост. на учет _____ от _____ выдано _____

Наименование товара	Кол-во	Цена	Сумма

Наличие кассового чека обязательно

Итого _____ Подпись _____

Рисунок 2.3— Бланк товарного чека

Був обраний алгоритм моделювання сезонності для прогнозування даних в магазині автозапчастин. Вибір пав на цей алгоритм, тому, що на відміну від інших алгоритмів прогнозування даних, він показує мінімальну похибку при розрахунках та в кінцевому результаті отримані значення кількості необхідного товару який має зберігатися на складі.

Вибраний алгоритм прогнозування даних буде пропонувати до закупівлі той чи інший товар, в залежності від сезонності та інших параметрів, тим самим зменшує витрати на зберігання зайвого товару, та збільшить прибуток магазину скоротив аналітичний відділ та своєчасно задовольняючи потреби покупців.

Розробка програмного забезпечення, дозволить значно скоротити час на обробку замовлення, так як пошук товарів буде проводиться в електронних каталогах і прайс-листах, а замовлення буде формуватися в електронній формі, автоматично визначаючи вартість замовлення. Форма звітності для бухгалтерії також буде формуватися автоматично на підставі таблиці бази даних про куплений товар.

2.3.4 Вимоги до системи

Підсистема “Адміністратор магазинів” призначена для покращення організації обліку товару та ефективного продажу.

Функціональні можливості:

- функціонал даної програми дозволяє оперативно і точно задовольняти потреби покупця;
- у режимі реального часу регулювати попит і потребу в товарі;
- вирішувати складські та логістичні завдання;
- формувати звіти по товарообігу;
- надавати дані для бухгалтерського обліку;
- планування діяльності підприємства.

Задача розробки програмного забезпечення полягає в необхідності продавця та адміністратора вести електронний облік товару, також програмне забезпечення дозволить скоротити час на обробку замовлення клієнта та формування звітності для бухгалтерії.

2.4 Функціональна задача.

2.4.1 Склад функціональних завдань системи

Технічне забезпечення підприємства включає в себе кілька ПК, які об'єднані в одну мережу, з'єднані через сервер, який зберігає базу даних.

На вхід в підприємство надходять накладні про товар, замовлені зі складу постачальника на склад магазину, інформацію про яких в свою чергу вносить адміністратор в базу даних. Проміжним документом є накладна видана покупцеві на придбання товару, звіт з якої так само заноситься в загальну базу. Вихідним документом є заявка постачальнику на необхідні товари для наповнення складу

Для подальшої реалізації DFD діаграми, була створена таблиця описання функцій схем даних

Таблиця 2.1 Описання функцій схем даних

№	Функція	Елемент	Опис
1	Підсистема магазину з продажу автозапчастин	Облік товару (автозапчастин)	Завдання передбачає врахування даних про закупівлю і продаж товару.
		Облік продаж	Завдання передбачає облік продаж магазину.
2	Облік товару (автозапчастин)	Облік про закупівлю товару.	Функція передбачає врахування товару, що купується і його постачальників (Назва деталі, марка, тип кузова, рік, тип двигуна, назва товару, виробник товару)
		Облік про продаж товару.	Функція передбачає загальну інформацію про товар і облік продавців (Прізвище клієнта, назва, марка, назва товару, дата)
3	Облік продаж	Облік про продаж товару.	Функція передбачає загальну інформацію про

			товар і облік продавців (Найменування, Опис, Наявність, Вартість продажу) (Код продавця, ПІБ, Стаж).
		Облік клієнтів	Функція передбачає список клієнтів магазину (Ім'я, прізвище, по батькові, номер телефону.)
4	Облік товару, що купується	Складання номенклатури товару, що купується	Функція передбачає перелік товару, що купується (Назва деталі, марка, тип кузова, рік, тип двигуна, назва товару, виробник товару)
5	Облік товару, що продається	Складання номенклатури товару, що продається	Функція передбачає перелік товару, що продається. (Назва деталі, марка, тип кузова, рік, тип двигуна, назва товару, виробник товару)
6	Облік клієнтів	-	Функція передбачає введення даних загальної інформації про клієнтів (Ім'я, прізвище, по батькові, номер телефону)

2.4.2 Вибір типових проектних рішень

Для реалізації програмного забезпечення підсистеми «Адміністратор магазинів» здійснювався вибір меду двома об'єктно-орієнтованими мовами C # і Java.

C # і Java - дві мови програмування, розвиваючих мову програмування C ++, з синтаксисом, багато в чому спадщиною синтаксис C ++, і створених багато в чому в умовах конкуренції, і, внаслідок цього, що володіють певним схожістю, а також мають і ряд відмінностей.

1. Об'єктні засоби.

Обидві мови - об'єктно-орієнтовані, з синтаксисом, успадкованим від C ++, але значно переробленим. Код і дані можуть описуватися тільки всередині класів.

Інкапсуляція.

В Java модифікатор `protected` в описі, крім доступу з класів-нащадків, надати їм доступ з усіх класів, що входять в той же пакет, що і клас-власник.

C # для об'єктів, які повинні бути видні в межах збірки (приблизний аналог пакету Java) введений окремий модифікатор `internal` (аналог `default` в Java), а `protected` зберігає свій початковий сенс, взятий з C ++ - доступ тільки з класів-нащадків. Допускається комбінувати `internal` і `protected` - тоді вийде область доступу, відповідна `protected` в Java.

Внутрішні класи.

Обидві мови дозволяють визначити клас всередині класу.

В Java внутрішні класи використовуються для емуляції замикань. Внутрішні класи Java мають доступ до нестатичних членів батьківського класу, тобто «знають про `this`»; крім того, всередині методів можна визначати локальні класи, які мають доступ по читанню до локальних змінних, і безіменні (анонімні) локальні класи, які фактично дозволяють створювати екземпляри об'єктів і інтерфейсів, що перекривають методи свого класу, безпосередньо в місці їх використання. Таким чином, зникає необхідність в спеціальному типі і синтаксичної підтримки для подій, але сам код, який створює обробники, кілька більш складний для розуміння. Зокрема, складніше стають області видимості змінних.

У C # є замикання і лямбда. Підхід C # більше нагадує C ++: внутрішні класи в C # мають доступ тільки до статичних членів зовнішнього класу, а для доступу до нестатичних членів потрібно явно вказувати екземпляр зовнішнього класу. Локальні внутрішні класи в C # і не підтримуються.

Java старше, ніж C # і побудована на великий і активної користувальницької базі, ставши *lingua franca* в багатьох сучасних областях інформатики, особливо таких, де задіяні мережі. Java домінує в курсах

програмування американських університетів і коледжів, і літератури по Java сьогодні набагато більше, ніж по C #. Зрілість і популярність Java привели до більшої кількості бібліотек і API на Java (багато з яких відкриті), ніж на C #.

З огляду на всі переваги і недоліки мови Java і мови C # для створення програмного забезпечення підсистеми «Адміністратор магазинів» була обрана мова програмування Java.

Програмне забезпечення підсистеми «Адміністратор магазинів» створено в середовищі розробки програмного забезпечення IntelliJ Idea, так як це середовище програмування дозволяє створювати додатки на мові програмування Java.

2.4.3 Технічне забезпечення

Автоматизована система повинна працювати на IBM-сумісних персональних комп'ютерах.

Мінімальна конфігурація комп'ютера:

- тип процесора – Intel Core Duo та вище;
- частота – 1,5 Гц;
- оперативна пам'ять - 2 Gb та вище;
- об'єм жорсткого диска - не менш 80 Гб;
- джерело безперебійного живлення CyberPower UT650E;
- комп'ютерна миша Sven RX-112 USB Black ;
- клавіатура Gemix KB-160;
- монітор 21.5" AOC e2270Swn Black;
- маршрутизатор TP-LINK TL-R860;
- мережеві кабелі;
- принтер HP LaserJet Pro M102a;

2.5 Постановка задачі

На основі проведеного аналізу предметної області, визначення проблем що існують, існуючих рішень та можливих концепцій необхідно розробити підсистему, що матиме наступний функціонал:

- прогнозувати закупівельний товар
- прогнозувати закупівлю товару за параметрами сезонності
- дозволяти оперативно і точно задовольняти потреби покупця
- у режимі реального часу регулювати попит і потребу в товарі
- вирішувати складські та логістичні завдання
- формувати звіти по товарообігу
- вивід друку звіту, або накладної
- додавання необхідної інформації в базу даних
- надавання даних для бухгалтерського обліку

3 Розробка технічного проекту підсистеми “Адміністратор магазинів”

3.1.1 Діаграма варіантів використання

У попередніх розділах були розглянуті предметна область, задачі і вимоги, що ставляться до задачі. Грунтуючись на цих даних, можна перейти до проектування і реалізації поставлених питань. Перший етап – це побудова діаграми варіантів використання.

Діаграми варіантів використання описують функціональне призначення системи або те, що система повинна робити. Розробка діаграми переслідує такі цілі:

- визначити спільні кордони і контекст модельованої предметної області;
- сформулювати загальні вимоги до функціонального поведінки проектованої системи;
- розробити вихідну концептуальну модель системи для її подальшої деталізації у формі логічних і фізичних моделей;
- підготувати вихідну документацію для взаємодії розробників системи з її замовниками і користувачами.

На рисунку 3.1 представлена діаграма варіантів використання

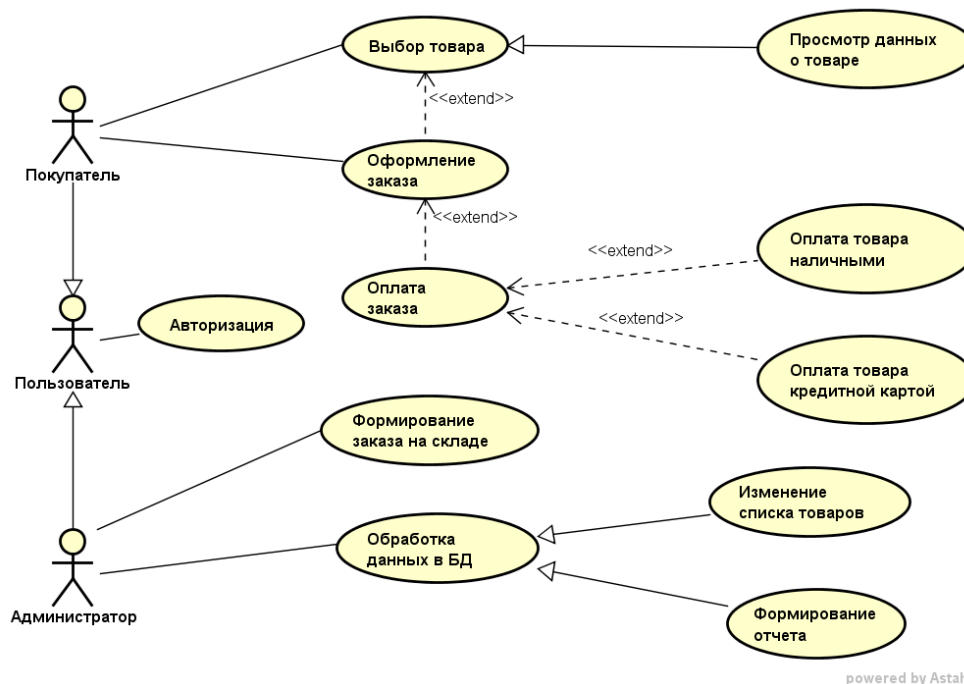


Рисунок 3.1 – Діаграма варіантів використання

3.1.2 Загальносистемні проектні рішення. Побудова функціональної моделі

Другий етап – це побудова діаграм потоків даних (DFD). Вони дозволять виявити сутності, що приймають участь в функціонуванні підсистеми, процеси які відбуваються в підсистемі і інформацію, що переходить від користувачів до підсистеми і навпаки, змінюючись при цьому.

Діаграма потоків даних нульового рівня приведена на рисунку 3.2.

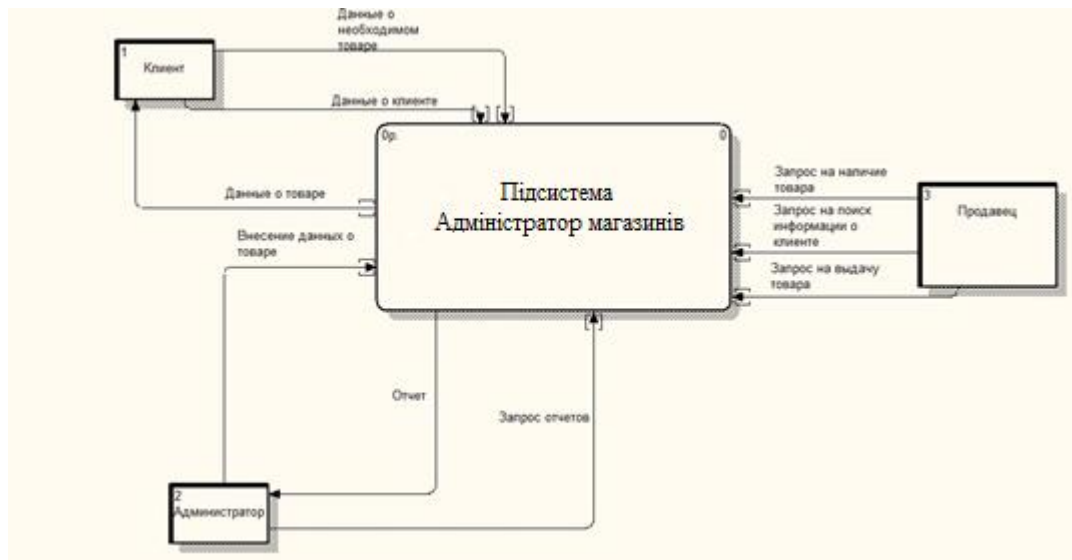


Рисунок 3.2 – Діаграма потоків даних нульового рівня

Проведемо декомпозицію основного процесу підсистеми «Адміністратор магазинів» і визначимо процеси, що реалізуватимуть окремі функції, необхідні для вирішення основної задачі. Діаграма потоків даних 1-го рівня показана на рисунку 3.3.

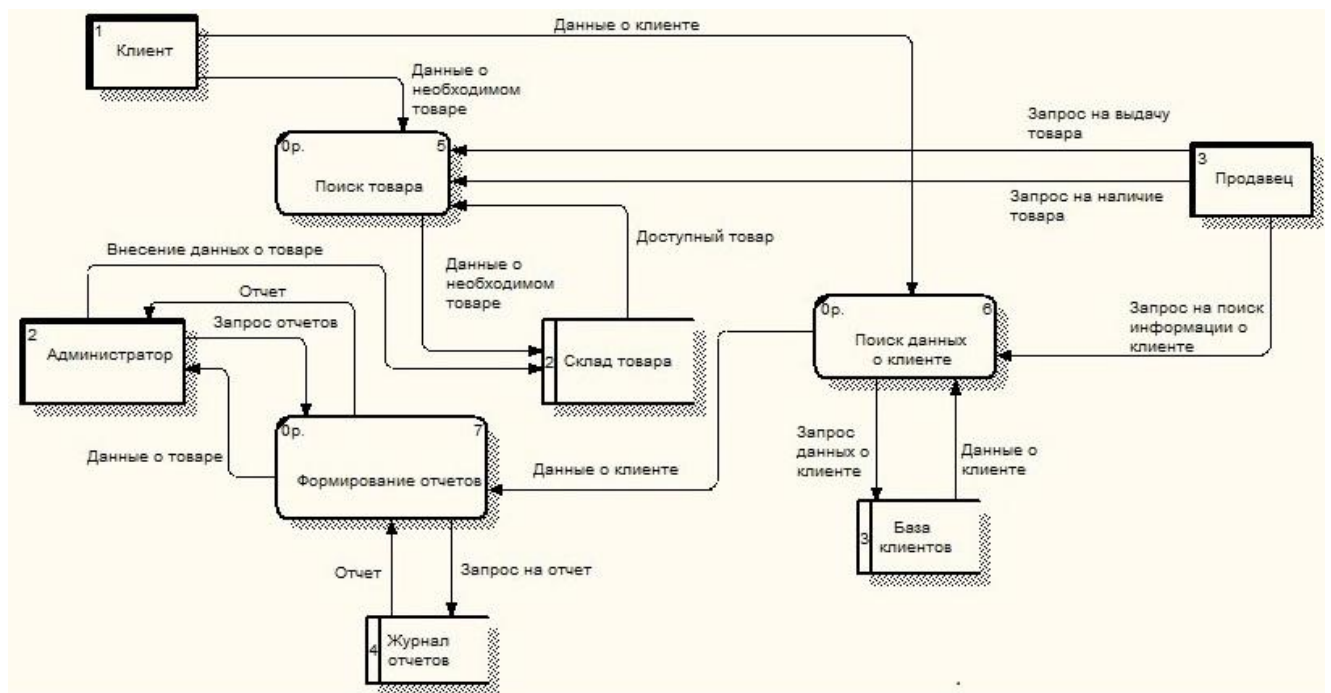


Рисунок 3.3 – Діаграма потоків даних першого рівня

На таблиці 3.1 представлений опис та атрибути DFD діаграм нульового та першого рівня

Потік даних 0 рівня	Потік даних 1 рівня	Атрибути
Клієнт	Дані о необходимом товарі	Назва деталі, марка, тип кузова, рік, тип двигуна, назва товару, виробник товару
	Дані о клієнті	Ім'я, прізвище, по батькові, номер телефону
	Данні о товарі	Назва деталі, марка, тип кузова, рік, тип двигуна, назва товару, виробник товару
Продавець	Запит на наявність товару	Назва деталі, марка, тип кузова, рік, тип двигуна, назва товару, виробник товару

	Запит на пошук інформації про клієнта	Ім'я, прізвище, по батькові, номер телефону.
	Запит на видачу товару	Назва деталі, марка, тип кузова, рік, тип двигуна, назва товару, виробник товару
Адміністратор	Внесення даних о товарі	Назва деталі, марка, тип кузова, рік, тип двигуна, назва товару, виробник товару
	Звіт	Прізвище клієнта, назва та марка автомобіля, назва товару, дата
	Запит звіту	Прізвище клієнта, назва та марка автомобіля, назва товару, дата

- 3.2 Рішення з організаційного забезпечення.

З системою інтеграції повинні працювати наступні підрозділи:

- Керуючий персонал;
- Амдіністрація;
- Бухгалтерія;
- Відділ продажу;

Штат структурних підрозділів автомагазину при впровадженні даної системи спрощується в підрозділі бухгалтерія, шляхом поєднання головного бухгалтера, бухгалтера та касира в одного працівника.

До обов'язків директора, адміністратора, продавців-консультантів і співробітника бухгалтерії, необхідно внести застосування системи, при виконанні відповідних функцій, згідно розробленої інструкції користувача (додаток Б).

– 3.3 Рішення з інформаційного забезпечення

3.3.1 Опис інформаційного забезпечення

Склад інформаційного забезпечення.

Інформаційне забезпечення системи інтеграції ґрунтується на єдиній базі даних автомагазина, яка повинна бути заповнена необхідними для системи таблицями.

Організація інформаційного забезпечення: архітектурою створеної бази даних є «клієнт-сервер». Всі дані зберігаються на сервері баз даних. За допомогою програмного додатку здійснюється ввід інформації, копіюванням з електронних документів або з клавіатури.

Опис організації інформаційної бази:

Розробимо структуру бази даних. Розробка проводиться в середовищі MySQL Workbench 6.3 та середовищі розробки IntelliJ IDEA 2017.2.5.

3.3.2 Побудова логічної моделі даних.

В попередньому підрозділі проведений аналіз системи з метою виявлення її процесів та потоків даних, що змінюються проходячи через процеси. Ґрунтуючись на отриманих даних, потрібно визначити інформацію, як буде зберігатись для забезпечення роботи програми. Це і є задача розділу.

Логічна модель - це загальний погляд на дані, модель даних в певній предметній області. Інколи її називають концептуальною моделлю даних. Такою моделлю і є модель типу сутність-зв'язок.

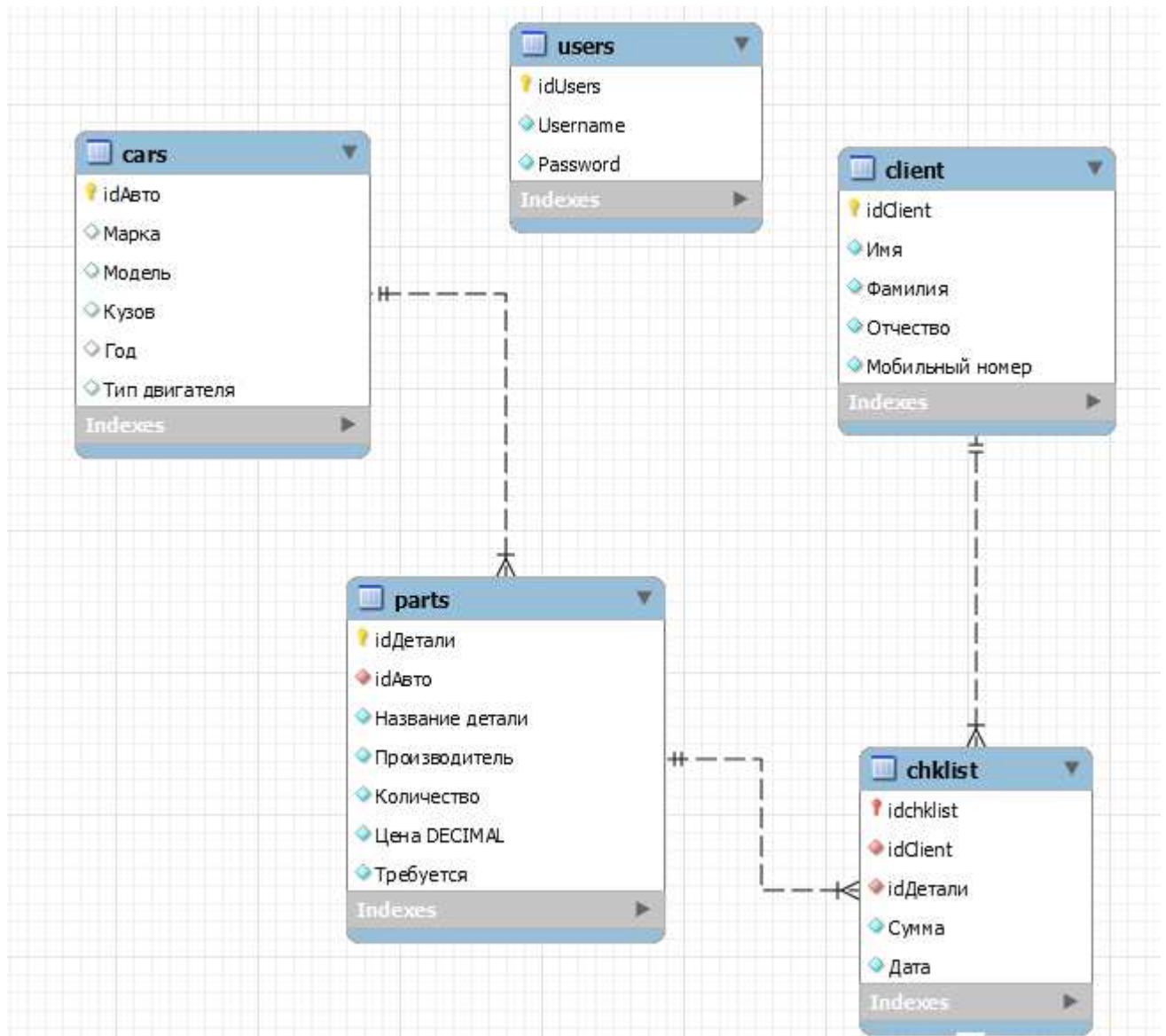


Рисунок 3.4 – Логічна модель даних

3.3.3 Побудова фізичної моделі даних

Наступний етап роботи – побудова фізичної моделі даних. В ній визначається тип даних для кожного поля в таблицях даних, а також ширина кожного поля. Кількість таблиць в фізичній моделі може бути більшою. Це пояснюється тим, що деякі дані, логічно поєднані, можуть фізично розноситись для зменшення часу обробки даних, зростання швидкості доступу до потрібної інформації. На рисунку 3.5 представлена фізична модель даних.

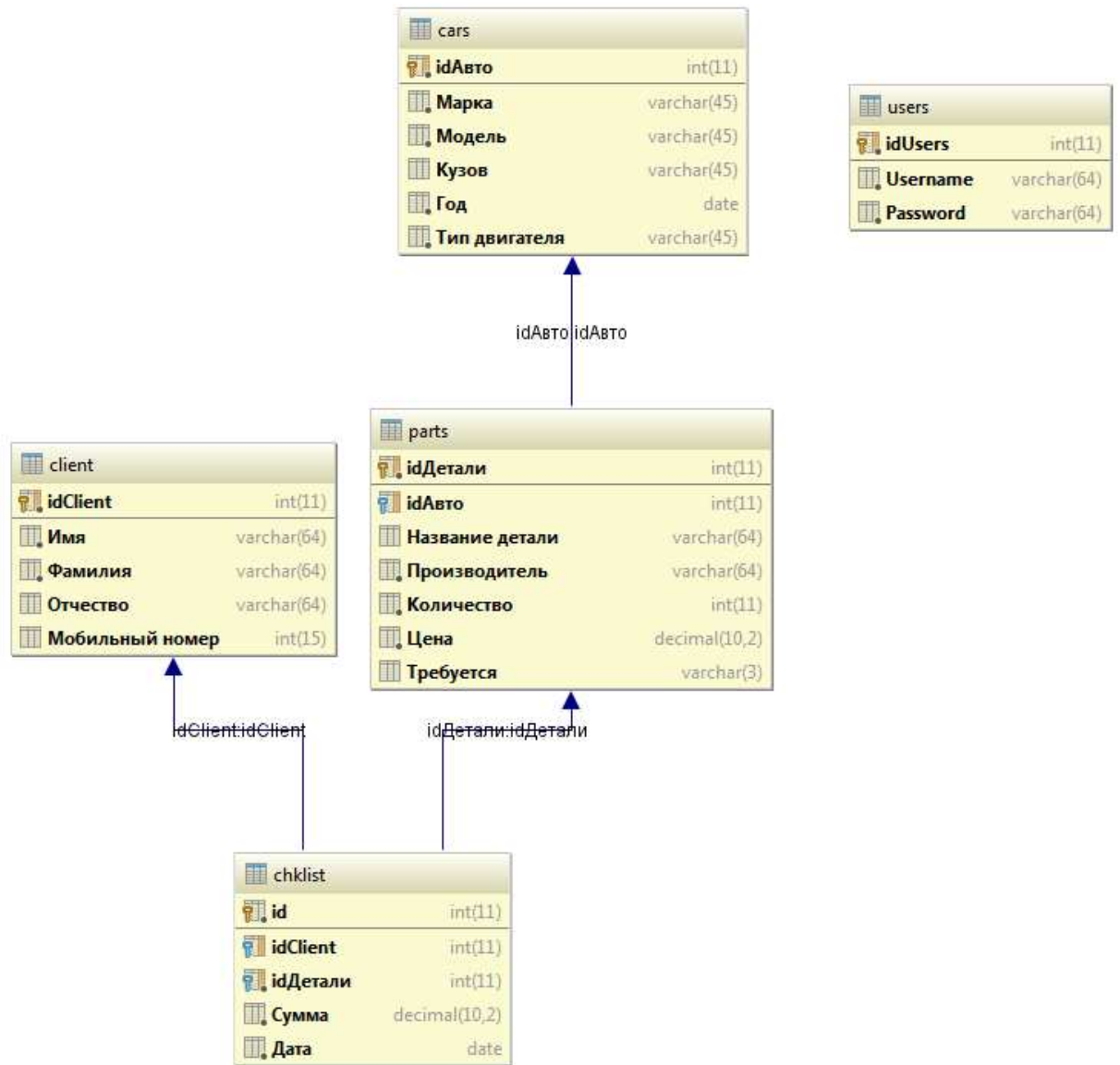


Рисунок 3.5 – Фізична модель даних

На таблиці 3.2 Предсталений опис фізичної моделі даних

Назва поля	Тип даних	Розмір	Ключ
idClient	int	11	Primary key
Имя	varchar	64	AK
Фамилия	varchar	64	
Отчество	varchar	64	
Мобильный номер	int	15	

Таблиця 3.2 - Таблица “client”

Назва поля	Тип даних	Розмір	Ключ
idUsers	int	11	Primary key
Username	varchar	64	
Password	varchar	64	

Таблиця 3.3 - Таблиця “users”

Назва поля	Тип даних	Розмір	Ключ
id	int	11	Primary key
idClient	int	11	Foreign key
idДетали	int	11	Foreign key
Сумма	decimal	10,2	
Дата	date		

Таблиця 3.4 - Таблиця “chklist”

Назва поля	Тип даних	Розмір	Ключ
idДетали	int	11	Primary key
idАвто	int	11	Foreign key
Название детали	varchar	64	
Производителе ль	varchar	64	
Количество	Int	11	
Цена	decimal	10,2	
Требуется	varchar	3	

Таблиця 3.5 - Таблиця “parts”

Назва поля	Тип даних	Розмір	Ключ
idАвто	int	11	Primarykey
Марка	varchar	45	
Модель	varchar	45	
Кузов	varchar	45	
Год	date		
Тип двигателя	varchar	45	

Таблиця 3.6 - Таблиця “cars”

Перелік таблиці, які необхідно використати з бази даних надано в рисунках 3.6-3.10.

На рисунку 3.6 представлена таблиця “cars”, та перелік полів до яких заносяться данні про марку, модель, кузов, рік випуску та тип двигуна автомобіля.

Column	Type	Nullable	Character Set	Collation	Privileges	Comments
◇ idАвто	int(11)	NO			select,insert,update,references	
◇ Марка	varchar(45)	NO	utf8	utf8_general_ci	select,insert,update,references	
◇ Модель	varchar(45)	NO	utf8	utf8_general_ci	select,insert,update,references	
◇ Кузов	varchar(45)	YES	utf8	utf8_general_ci	select,insert,update,references	
◇ Год	date	NO			select,insert,update,references	
◇ Тип двигателя	varchar(45)	NO	utf8	utf8_general_ci	select,insert,update,references	

Рисунок 3.6 – Таблиця “cars”

На рисунку 3.7 представлена таблиця “chklist”, та перелік полів до яких заносяться данні необхідні для заповнення накладних

Column	Type	D	Nullable	Character Set	Privileges	Extra	Comments
◇ id	int(11)		NO		select,insert,update,references	auto_increment	
◇ idClient	int(11)		YES		select,insert,update,references		
◇ idДетали	int(11)		NO		select,insert,update,references		
◇ Сумма	decimal(10,2)		NO		select,insert,update,references		
◇ Дата	date		NO		select,insert,update,references		

Рисунок 3.7 – Таблиця “chklist”

На рисунку 3.8 представлена таблиця “client”, та перелік полів до яких заносяться данні про ім’я, прізвище, по батькові та номер мобільного телефона клієнта магазину

Column	Type	Nullable	Character Set	Collation	Privileges	Extra	Comments
idClient	int(11)	NO			select,insert,update,references	auto_increment	
Имя	varchar(64)	NO	utf8	utf8_general_ci	select,insert,update,references		
Фамилия	varchar(64)	NO	utf8	utf8_general_ci	select,insert,update,references		
Отчество	varchar(64)	YES	utf8	utf8_general_ci	select,insert,update,references		
Мобильный номер	int(15)	YES			select,insert,update,references		

Рисунок 3.8 – Таблица “client”

На рисунку 3.9 представлена таблиця “parts”, та перелік полів до яких заносяться данні про ідентифікатор авто, назву деталі, виробника деталі, кіллькість товару на складі, ціну на товар та інформацію про те, чи потребує деталь замовлення

Column	Type	Nullable	Character Set	Collation	Privileges	Extra	Comments
idДетали	int(11)	NO			select,insert,update,references	auto_increment	
idАвто	int(11)	YES			select,insert,update,references		
Название детали	varchar(64)	YES	utf8	utf8_general_ci	select,insert,update,references		
Производитель	varchar(64)	NO	utf8	utf8_general_ci	select,insert,update,references		
Количество	int(11)	NO			select,insert,update,references		
Цена	decimal(10,2)	NO			select,insert,update,references		
Требуется	varchar(3)	YES	utf8	utf8_general_ci	select,insert,update,references		

Рисунок 3.9 – Таблица “parts”

На рисунку 3.10 представлена таблиця “users”, та перелік полів до яких заносяться данні, логін та пароль користувачів системи.

Column	Type	Nullable	Character Set	Collation	Privileges	Extra	Comments
idUsers	int(11)	NO			select,insert,update,references	auto_increment	
Username	varchar(64)	NO	utf8	utf8_general_ci	select,insert,update,references		
Password	varchar(64)	NO	utf8	utf8_general_ci	select,insert,update,references		

Рисунок 3.10 – Таблица “users”

3.4 Рішення з технічного забезпечення

3.4.1 Склад та структура технічного забезпечення системи

Технічне забезпечення проектованої системи охоплює повний набір апаратних засобів і потребує вводу додаткового технічного забезпечення. Технічне забезпечення складається з робочих станцій адміністратора, директора та головного бухгалтера. В використання адміністратора та головного бухгалтера надається додаткове технічне забезпечення у вигляді принтеру. Для з'єднання робочих станцій в мережу та зв'язку з серверною частиною використовуються комутатори. Серверна частина складається з серверу бази даних магазину. Схематично структуру технічного забезпечення можна зобразити наступним чином (Рисунок 3.11).

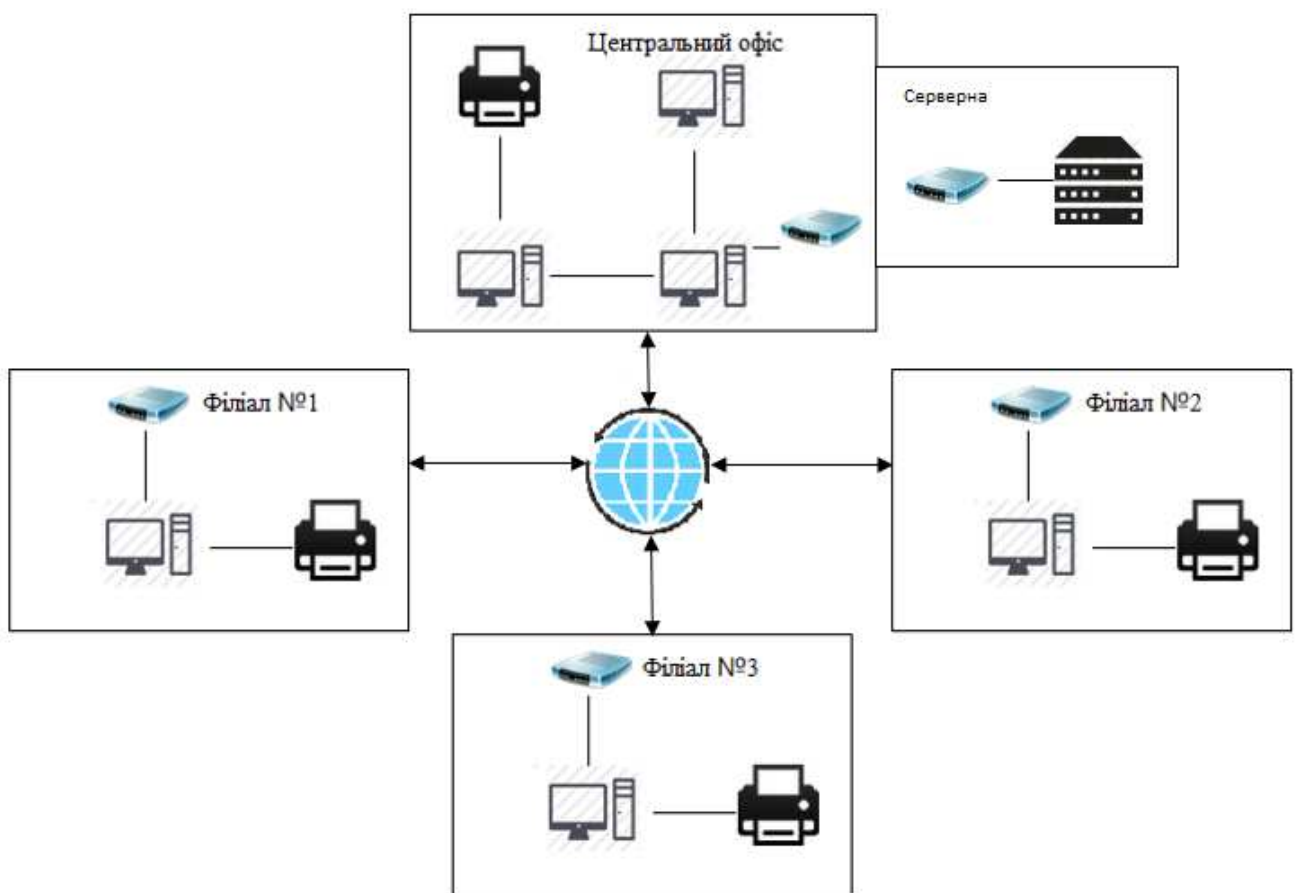


Рисунок 3.11 – Структурна схема технічного забезпечення

– 3.4.2 Опис комплексу технічних засобів

В якості сервера баз даних пропонується встановити комп'ютер IBM Intel core 2 duo 2.0/3 Gb з SATA-RAID 150 і мінімум вінчестер з дисковим простором в 500 Гб. Для сервера необхідно встановити додатковий блок безперервного живлення з можливістю стабілізації напруги і автономністю на половинній

потужності не менше 2-х годин. Сервер баз даних підключити до мережі через маршрутизатор, що встановлений в серверній. Також пропонується замінити 2 комп'ютера IBM Intel Celeron 2.0, та встановити додатково ще один комп'ютер на робоче місце продавця-консультанта і підключити їх до локальної мережі через концентратор.

3.5 Рішення з програмного забезпечення

3.5.1 Розробка статичної моделі програмного забезпечення

Статична модель поведінки програмного забезпечення описана за допомогою схеми взаємодії програмних модулів. Діаграма взаємодії призначена для моделювання відносин між об'єктами (ролями, класами, компонентами) системи в рамках одного прецеденту.

Даний вид діаграм відображає наступні аспекти проектованої системи:

- обмін повідомленнями між об'єктами (у тому числі в рамках обміну повідомленнями зі сторонніми системами);
- обмеження, що накладаються на взаємодію об'єктів;
- події, які ініціюють взаємодії об'єктів.

Схема взаємодії програмних модулів зображена на рисунку 3.12

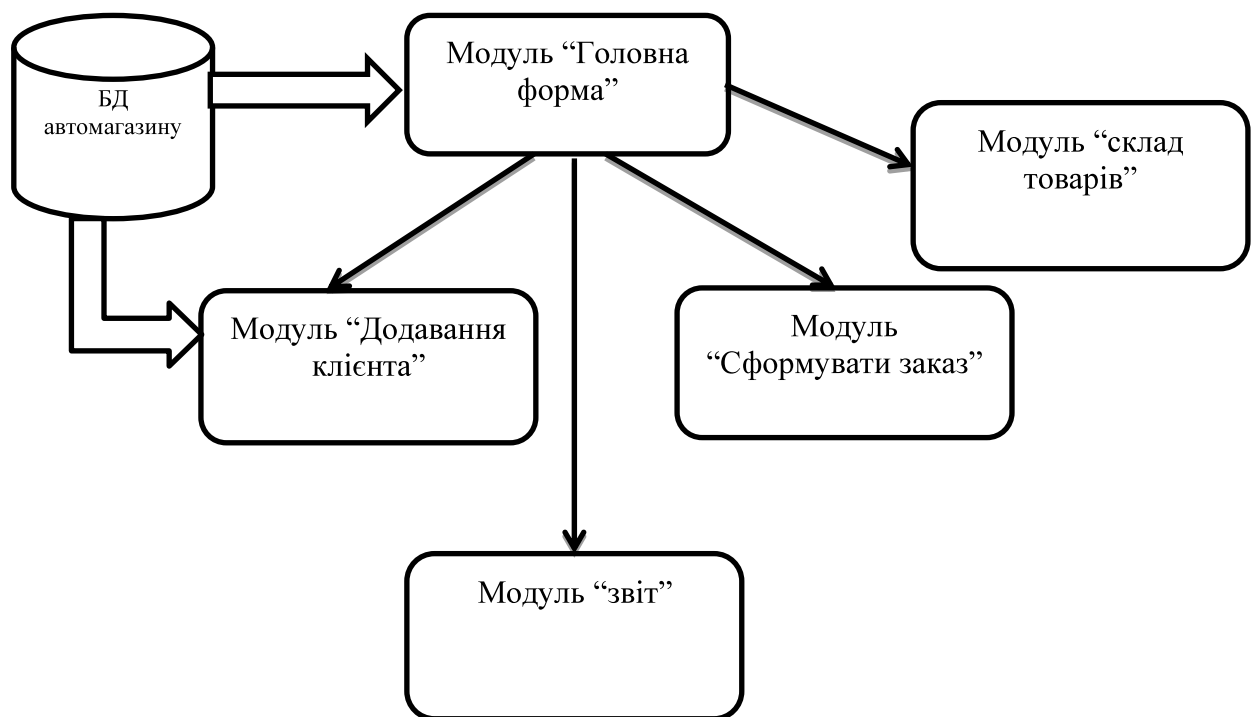
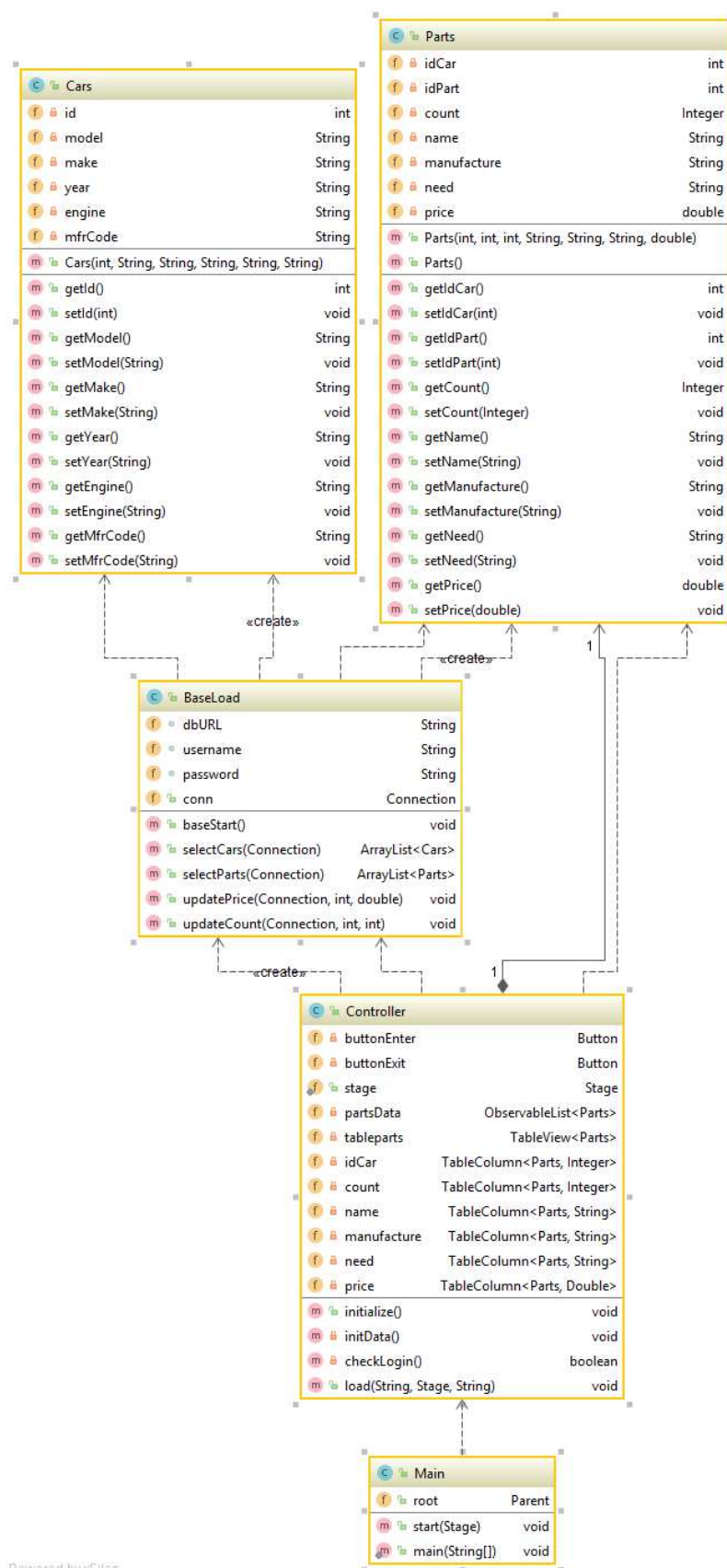


Рисунок 3.12 – Схема взаємодії програмних модулів

3.5.2 Діаграма класів

Діаграма класів (class diagram) служить для представлення статичної структури моделі системи в термінології класів об'єктно-орієнтованого програмування. Діаграма класів може відбивати, зокрема, різні взаємозв'язки між окремими сутностями предметної області, такими як об'єкти і підсистеми, а також описувати їх внутрішню структуру і типи відносин.



Powered by yFiles

Рисунок 3.13 – Діаграма класів

3.5.3 Розробка проекту інтерфейсу

Розроблювана система має звичний графічний інтерфейс користувача, інтуїтивного розуміння, що виключить потребу спеціального навчання персоналу.

Одночасно з відкриттям програми з'являється вікно авторизації, яке пропонує ввести логін та пароль, або вийти із програми(рисунок 3.14).

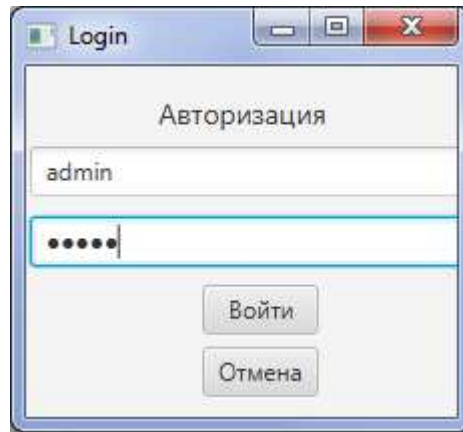


Рисунок 3.14 – Вікно авторизації

Якщо було введено коректний логін та пароль, програма автоматично перейде до головного меню (рисунок 3.15).

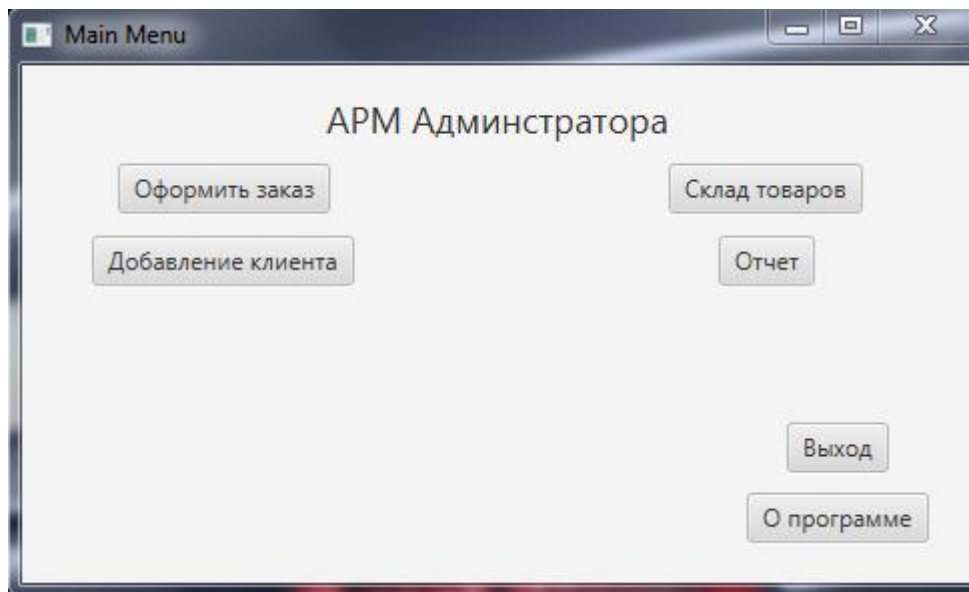


Рисунок 3.15 – Головне меню

Натиснувши кнопку головного меню “Оформить заказ”, відкривається вікно за допомогою якого можна сформувати заказ для покупця, та надрукувати його (рисунок 3.16).

The screenshot shows a software window titled "Ordering" with a subtitle "Оформление заказа". At the top right is a button "Печать накладной". Below the title, there are two input fields: "Покупатель:" with a dropdown menu showing "Иванов" and "Дата:" with a text box showing "29.05.2017" and a calendar icon. The main area contains a table with five columns: "Автомобиль", "Товар", "Количество", "Цена", and "Сумма". The first row of the table has the following values: "Opel,Astra,1999,1.6 16" in the "Автомобиль" column, "Амортизатор передн" in the "Товар" column, "2" in the "Количество" column, "255,55" in the "Цена" column, and "511,1" in the "Сумма" column. Below the table, there is a label "Всего к оплате:" and a button "Отмена".

Автомобиль	Товар	Количество	Цена	Сумма
Opel,Astra,1999,1.6 16	Амортизатор передн	2	255,55	511,1

Рисунок 3.16 – Вікно оформлення замовлення

Натиснувши кнопку головного меню “Добавление клиента” відкривається вікно за допомогою якого, можна додати нового клієнта, якщо його ще не має в базі.(рисунок 3.17).

The screenshot shows a software window titled "New client" with a subtitle "Добавление клиента". It contains four input fields: "Имя" with "Василий", "Фамилия" with "Иванов", "Отчество" with "Иванович", and "Номер телефона" with "+380951234567". At the bottom, there are two buttons: "Добавить" and "Отмена".

Имя	Василий
Фамилия	Иванов
Отчество	Иванович
Номер телефона	+380951234567

Рисунок 3.17 – Вікно додавання нового клієнта

Натиснувши кнопку головного меню “Склад товаров” відкривається база даних всіх товарів, які є на складі, та тих що потрібно замовити.(рисунок 3.18).

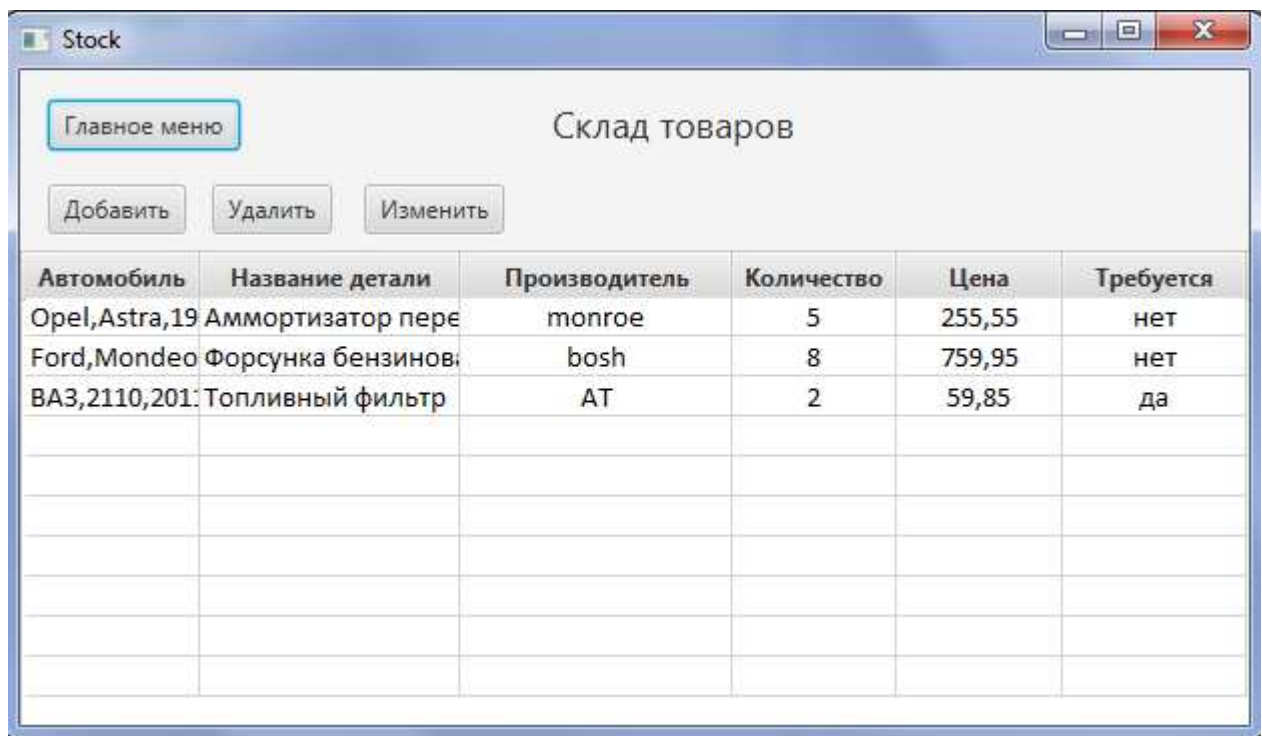
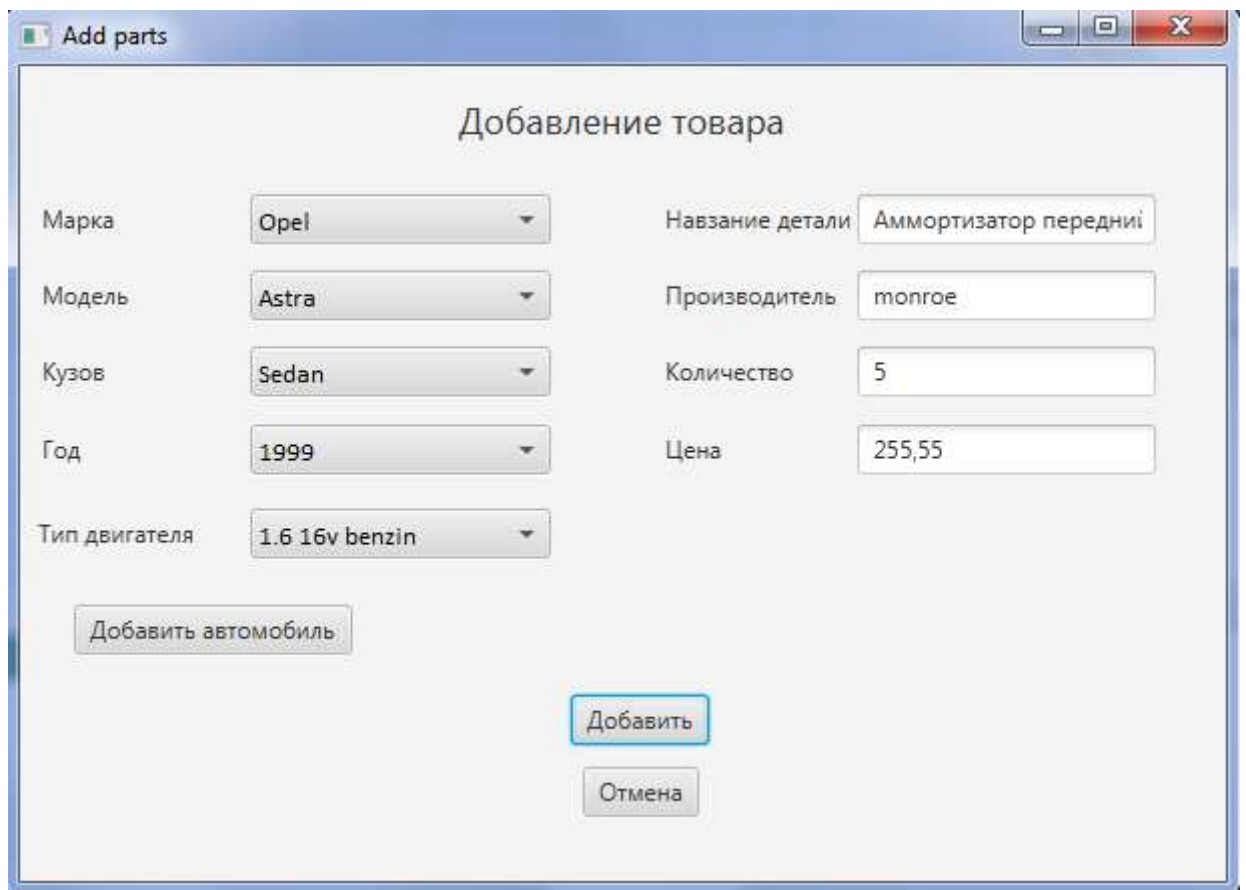


Рисунок 3.18 – Склад товарів

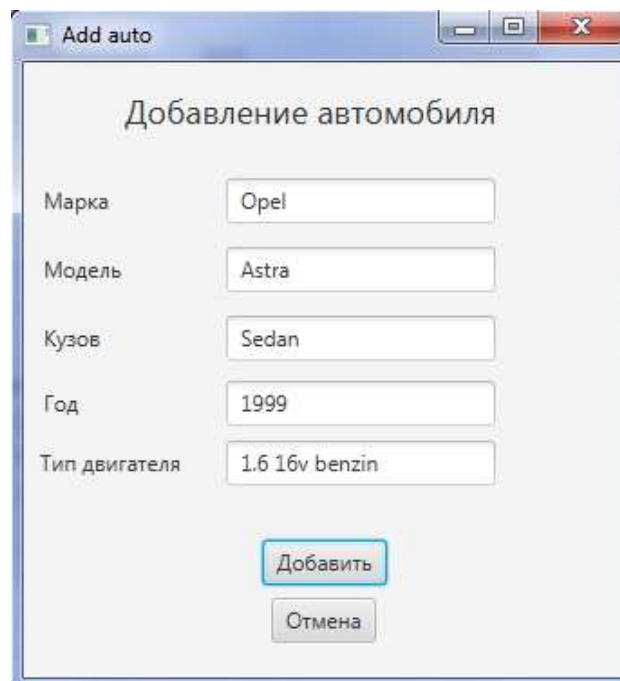
Якщо в вікні “Склад товарів” натиснути кнопку “Добавить” відкриться вікно, за допомогою якого можна додати новий товар на склад, або якщо обрати товар зі списку, та натиснути кнопку “Изменить”, то відкриться форма редагування товару (рисунок 3.19).



The screenshot shows a window titled "Add parts" with a subtitle "Добавление товара". It contains two columns of form fields. The left column has dropdown menus for "Марка" (Opel), "Модель" (Astra), "Кузов" (Sedan), "Год" (1999), and "Тип двигателя" (1.6 16v benzin). The right column has text input fields for "Название детали" (Амортизатор передний), "Производитель" (mongoe), "Количество" (5), and "Цена" (255,55). At the bottom left is a button "Добавить автомобиль", and at the bottom center are two buttons: "Добавить" (highlighted with a blue border) and "Отмена".

Рисунок 3.19 Вікно додавання та редагування товару

В вікні “Добавление товара”, можна натиснути кнопку “Добавить автомобиль”, після чого відкриється вікно додавання нового автомобіля в базу(рисунок 3.20).



The screenshot shows a window titled "Add auto" with a subtitle "Добавление автомобиля". It contains a single column of text input fields for "Марка" (Opel), "Модель" (Astra), "Кузов" (Sedan), "Год" (1999), and "Тип двигателя" (1.6 16v benzin). At the bottom are two buttons: "Добавить" (highlighted with a blue border) and "Отмена".

Рисунок 3.20 – Додавання нового автомобіля в базу

Натиснувши кнопку головного меню “Отчет” відкриється вікно формування звіту(рисунок 3.21).

Report

Отчет

С 28.05.2017 По 29.05.2017 За весь период

Покупатель Иванов

Автомобиль

Товар

Сформировать отчет Печать отчета

Отмена

Рисунок 3.21 – Формування звіту

4. Розробка робочої документації

Підсистема “Адміністратор магазинів” дозволяє виконувати наступні функції:

- дозволяти оперативно і точно задовольняти потреби покупця
- у режимі реального часу регулювати попит і потребу в товарі
- вирішувати складські та логістичні завдання
- формувати звіти по товарообігу
- вивід друку звіту, або накладної
- додавання необхідної інформації в базу даних
- надавання даних для бухгалтерського обліку

Впровадження розробленої підсистеми має на меті покращення ефективності роботи магазину та його персоналу, усунення можливих втрат інформації та помилок, забезпечить економію часу.

Розроблена підсистема має звичний графічний інтерфейс користувача, інтуїтивного розуміння, що виключить потребу спеціального навчання персоналу.

Для ефективного використання даної підсистеми застосовується технічне забезпечення яке описано в пункті 3.3.

При роботі з розробленим програмним забезпеченням варто керуватися документом “Інструкція користувача” (Додаток Б).

Спроектоване програмне забезпечення відповідає вимогам які викладені в технічному завданні (Додаток А).

Реалізовані модулі програмного забезпечення приведені в додатку В.

Основним засобом розробки програмного забезпечення системи є програмне середовище розробки IntelliJ IDEA 2017.2.5. Програмні модулі розроблені мовою програмування Java. В якості інструменту для ведення бази даних використовувалась СУБД MySQL Workbench 6.3.

1.1 5 Охорона праці

2.5.1 Вступ

Охорона праці — один з центральних інститутів трудового права. Він має виключно практичне значення. Недодержання вимог охорони праці створює небезпеку для здоров'я і життя працівників. Тому власники підприємств несуть сувору, у тому числі й кримінальну, відповідальність за порушення правил охорони праці.

Охорона праці, як соціально-економічне і юридичне поняття, — це система правових, соціально-економічних, організаційно-технічних, санітарно-гігієнічних та лікувально-профілактичних заходів та засобів, спрямованих на збереження здоров'я і працездатності людини в процесі праці (ст. 1 Закону "Про охорону праці"). Як правовий інститут, охорона праці — це комплекс правових норм.

Ст. 4 Закону "Про охорону праці" формулює принципи державної політики у сфері охорони праці. Ці принципи в силу вказаного їх визначення, слід кваліфікувати і як принципи інституту охорони праці, які підлягають застосуванню при відсутності відповідних конкретних правових норм, і як такі, що визначають напрямки подальшого розвитку інституту охорони праці, і як обов'язки держави, які вона взяла на себе і має додержувати при здійсненні своєї політики у майбутньому. З урахуванням викладеного, принципи, зазначені в ст. 4 Закону "Про охорону праці", мають певне регулятивне значення. У названій статті закріплені такі принципи:

1) Пріоритет життя і здоров'я працівників відносно результатів виробничої діяльності підприємства, повна відповідальність власника за створення безпечних і нешкідливих умов праці.

2) Комплексне вирішення завдань охорони праці на засадах національних програм з цих питань, а також з урахуванням інших напрямків економічної і соціальної політики, досягнень в галузі науки, техніки та охорони здоров'я.

3) Соціальний захист працівників, повне відшкодування шкоди особам, потерпілим від нещасних випадків на виробництві та професійних захворювань.

4) Установлення єдиних нормативів з охорони праці для всіх підприємств незалежно від форм власності і видів їх діяльності.

5) Використання економічних методів управління охороною праці, проведення політики пільгового оподаткування, що сприяє створенню безпечних та нешкідливих умов праці, участь держави в фінансуванні заходів з охорони праці.

6) Здійснення навчання населення, професійної підготовки та підвищення кваліфікації працівників з питань охорони праці.

7) Забезпечення координації діяльності державних органів, установ, організацій та об'єднань громадян, що вирішують різні проблеми охорони здоров'я, гігієни та безпеки праці, а також співробітництво.

Умови праці на робочому місці, безпека технологічних процесів, машин, механізмів, устаткування та інших засобів виробництва, стан засобів колективного та індивідуального захисту, що використовуються працівником, а також санітарно-побутові умови повинні відповідати вимогам нормативних актів про охорону праці.

Власник або уповноважений ним орган повинен впроваджувати сучасні засоби техніки безпеки, які запобігають виробничому травматизмові, і забезпечувати санітарно-гігієнічні умови, що запобігають виникненню професійних захворювань працівників.

На власника або уповноважений ним орган покладається систематичне проведення інструктажу (навчання) працівників з питань охорони праці, протипожежної охорони.

Охорона праці – один з центральних інститутів трудового права. Він має виключно практичне значення. Недодержання вимог охорони праці створює небезпеку для здоров'я і життя працівників. У свою чергу і ті, кого законодавець називає власником або уповноваженим ним органом, несуть сувору, у тому числі й кримінальну, відповідальність за порушення правил охорони праці.

3.5.2 Аналіз небезпечних і шкідливих факторів в магазині

Робота в магазині безпосередньо пов'язана з комп'ютерною технікою. З точки зору впливу небезпечних і шкідливих факторів при роботі з ПК можна виділити наступні:

- вплив електромагнітного й електростатичного випромінювань;
- вплив електричного струму;
- вплив шуму;
- недостатня освітленість робочого місця;
- нераціональне розташування устаткування і неправильна організація робочого місця;
- несприятливі умови мікроклімату;
- нераціональне розташування устаткування і неправильна організація робочого місця.

Вплив електромагнітного й електростатичного випромінювання. При роботі ПК виникають два типи випромінювань: електростатичне й електромагнітне. Навколо електростатично зарядженого монітора підвищується концентрація пилу. Електромагнітне випромінювання створюється магнітними котушками, що знаходяться біля цокольної частини ЕПТ.

Результати медичних досліджень показують, що електризований пил викликає запалення шкіри, а електромагнітне випромінювання приводить до порушення біологічних процесів (порушення сну, зниження кров'яного тиску, уповільнення серцевої діяльності, зміни складу крові).

Вплив електричного струму. Електричні установки, до яких належить все обладнання ЄОМ представляють для людини потенційну загрозу. Дія електричного струму може призвести до травми. Важливе значення для запобігання електричного травматизму є організація обслуговування комп'ютерної техніки.

Приміщення, де знаходяться робочі місця користувачів, належать до категорії приміщень без завищеної загрози, обладнання належить до класу до

1000В. Для запобігання електротравматизму досить застосувати захисне заземлення.

Вплив шуму. Шум – це звук, що заважає роботі, мові та негативно впливаючий на здоров'я людини. З точки зору впливу на людину, характеристиками шуму є інтенсивність, звуковий тиск, щільність. Для боротьби з шумом вдаються наступних заходів: збільшення звукоізоляції огорожуючих конструкцій, ущільнення дверей по периметру, зменшення шуму джерела шляхом застосування прокладки з еластичних матеріалів. Допустимий рівень шуму – 50 Дб.

Недостатня освітленість робочого місця. Освітлення – головний фактор створення нормальних умов праці. У приміщеннях з ЕОМ, як правило, застосовується природне бокове освітлення. Робочі кімнати і кабінети повинні мати природне освітлення. В інших приміщеннях допускається штучне освітлення.

У випадках, коли одного природного освітлення не вистачає, встановлюється комбіноване освітлення. При цьому додаткове штучне загальне освітлення застосовується не тільки в темний, але й у світлий період доби. Дopusкається використання місцевого штучного освітлення, при якому світильник встановлюється ліворуч від працюючого. Освітлення приміщення й устаткування повинно бути м'яким, без блиску.

Несприятливі умови мікроклімату. Мікрокліматичні параметри (температура, відносна вологість і швидкість руху повітря) впливають на самопочуття і здоров'я людини, а також на надійність роботи засобів обчислювальної техніки.

При підвищеній температурі і вологості в приміщенні відбувається перегрівання тіла, що погіршує самопочуття і працездатність. Варто враховувати, що ПК, які знаходяться в приміщенні, сприяють збільшенню температури.

Для комфортної роботи приймається температура $23 (\pm 2)^{\circ}\text{C}$ и $18 (\pm 2)^{\circ}\text{C}$ в теплий і холодний час року відповідно, при відносній вологості $55 (\pm 5)\%$.

Нераціональне розташування устаткування і неправильна організація робочого місця. Робоче місце – це частина простору, у якій користувач ПК здійснює трудову діяльність і проводить велику частину робочого часу.

Важливу роль грає планування робочого місця, яке повинне задовольняти вимогам зручності виконання робіт і економії енергії і часу користувача, зручності обслуговування ЕОМ, дотримання правил охорони праці. При плануванні робочого місця необхідно враховувати зони досяжності при розташуванні моніторів, клавіатури, маніпулятора типу «миша». Ці зони встановлюються на підставі антропологічних даних людського тіла.

4.5.3 Розрахунок системи штучного освітлення приміщення обчислювальної техніки магазину.

У приміщенні де розташована обчислювальна техніка необхідно організувати змішане освітлення, тобто сполучення природного і штучного освітлення. Природне освітлення створюється бічним освітленням через вікно. Штучне освітлення використовується при недостатньому природному освітленні. У даному приміщенні використовується загальне штучне освітлення.

Розрахунок його здійснюється по методу світлового потоку з урахуванням потоку, відбитого від стін і стелі.

Потрібно зробити розрахунок загального освітлення виробничого приміщення.

Дані для розрахунку:

- довжина $A = 8,0$ м;
- ширина $B = 4,0$ м;
- висота $H = 3$ м;
- висота робочої поверхні $h_p = 1$ м;
- світильник типу "УПМ-15";
- мінімальна освітленість лампи накаливання по нормах $E_{\min} = 100$ лк;
- коефіцієнт відображення стелі $S_{\pi}=50\%$, стін $S_c=50\%$, робочої поверхні $S_p=10\%$. Напруга мережі 220 В.

Відстань від стелі до робочої поверхні:

$$H_0 = H - h_p = 3 - 1 = 2 \text{ м}$$

Відстань від стелі до світильника:

$$h_c = 0,2 H_0 = 0,2 * 2 = 0,4 \text{ м}$$

Висота підвісу світильника над освітленою поверхнею:

$$h = H_0 - h_c = 2 - 0,4 = 1,6 \text{ м}$$

Висота підвісу світильника над підлогою:

$$H_n = h + h_p = 1,6 + 1 = 2,6 \text{ м}$$

Для досягнення найбільшої рівномірності освітлення приймаємо відношення $L / h = 1,5$.

Відстань між центрами світильників:

$$L = 1,5 h = 1,5 * 1,6 = 2,4 \text{ м}$$

Необхідна кількість світильників:

$$N = S / L^2 = (8 * 4) / (2,4)^2 = 32 / 5,76 = 5,56$$

Приймаємо кількість світильників: 5.

Індекс приміщення:

$$i = (A B) / (h (A + B)) = (8 * 4) / (1,6 (8 + 4)) = 32 / 19,2 = 1,67$$

Коефіцієнт використання світлового потоку $\eta = 0,50$

Світловий потік однієї лампи:

$$\Phi = (E_{\min} S K_3 Z) / (N \eta) = (100 * 32 * 1,3 * 1,15) / (5 * 0,50) = 1914 \text{ лм}$$

За знайденим світловим потоком з таблиці "Параметри ламп розжарювання загального призначення з розрахунковими напругами 130 и 220В" обираємо лампу Г125-135-150 потужністю 150 Вт., що має світловий потік 2280 лм, найбільш близький до розрахункового.

Фактична освітленість:

$$E_{\Phi} = E_{\min} \Phi_{\text{л}} / \Phi_{\text{р}} = 100 * 2280 / 1914 = 119,1 \text{ лк}$$

Загальна потужність:

$$P_{\text{общ}} = P_{\text{л}} N = 150 * 5 = 750 \text{ Вт} = 0,75 \text{ кВт}$$

5.5.4 Розробка заходів щодо зменшення впливу небезпечних і шкідливих факторів

Нормована освітленість робочого столу з ВДТ і ПЕОМ повинна бути в межах 300-500 лк. Допускається установка світильників місцевого освітлення для підсвічування документів. При цьому місцеве освітлення так само, як і загальне, не повинно створювати відблисків на екрані відеомоніторів. Зовнішня освітленість екрана монітора не повинна перевищувати 300 лк.

Бажано, щоб природне світло падало ліворуч, хоча допускається і таке розміщення робочого місця, коли віконні прорізи розташовані праворуч. Вікна повинні бути обладнані пристроями типу жалюзі, зовнішніх козирків і ін.

Зниження шуму, створюваного ПК, є дуже важливою задачею. Зниження шуму в джерелі випромінювання можна забезпечити застосуванням пружних прокладок між підставою ПК і опорною поверхнею. Як прокладки використовуються гума, повсть, пробка, різної конструкції амортизатори.

Під настільні шумливі апарати можна підкладати м'які килимки із синтетичних матеріалів, а під ніжки столів, на яких вони встановлені, прокладки з м'якої гуми, повсті товщиною 6 – 8 мм. Кріплення прокладок можливо шляхом приклейки їх до опорних частин.

Можливо також застосування звукоізолюючих кожухів, що не заважають технологічному процесові. Не менш важливим для зниження шуму в процесі експлуатації є питання правильного і своєчасного регулювання, змазування і заміни механічних вузлів шумливого устаткування.

Раціональне планування приміщення, розміщення устаткування є важливим чинником, що дозволяє знизити шум при існуючому устаткуванні ЕОМ. Машинний зал і приміщення для сервісної апаратури необхідно розташовувати удалині від шумливого і вібруючого устаткування.

Зниження рівня шуму, що проникає у виробниче приміщення ззовні, може бути досягнуто збільшенням звукоізоляції конструкцій, що обгороджують, ущільненням по периметру притворів вікон, дверей.

У такий спосіб для зниження шуму створюваного на робочих місцях внутрішніми джерелами, а також шуму, що проникає ззовні, потрібно:

- послабити шум самих джерел (застосування екранів, звукоізолюючих кожухів);
- знизити ефект сумарного впливу відбитих звукових хвиль (звуковбирні поверхні конструкцій);
- застосовувати раціональне розташування устаткування;
- використовувати архітектурно-планувальні і технологічні рішення ізоляцій джерел шуму.

До робочого столу теж пред'являються певні вимоги. Його конструкція повинна забезпечувати оптимальне розміщення на його площі використовуваного устаткування. Робочий стілець користувача необхідно розташовувати з таким розрахунком, щоб він дозволяв змінювати позу працюючого, а також був підйомно-поворотним. Це допоможе уникнути стомлення м'язів шийно-плечової області і спини.

Оптимальна відстань від очей до екрана монітора - 600-700 мм, але не ближче 500 мм з урахуванням розмірів алфавітно-цифрових знаків і символів. Недотримання цього параметра може призвести до швидкого розвитку стомлення і перевтоми зорового аналізатора і зі збільшенням тривалості роботи сприяти розвитку короткозорості.

Відстань між робочими столами з відеотерміналами повинне бути не менш 2,0 м, а відстань між бічними поверхнями відеомоніторів - не менш 1,2 м. Робочі місця з ВДТ і ПЕОМ у залах електронно-обчислювальних машин чи у приміщеннях із джерелами шкідливих виробничих факторів розміщаються в ізольованих кабінах з організованим повітрообміном.

6. 6.1 Вступ

Проблема захисту навколишнього середовища останнім часом стала однією з найбільш важливих в світі. Тривалий час вона не отримувала належної уваги, але зараз людство починають відчуватись негативні наслідки людської діяльності. Тільки оперативні заходи дозволять врятувати природне середовище, а разом з ним і суспільство, хоч не виключно, що деградація навколишнього середовища може бути в деяких моментах незворотною.

Охорона навколишнього середовища – це система мір, спрямована на збереження безпечного для існування живої і неживої природи навколишнього середовища, захисту життя і здоров'я населення від негативного впливу, обумовленого забрудненням навколишнього природного середовища. Забруднення природного середовища промисловими викидами впливає на людей, тварин, рослини, ґрунт, будинки і спорудження, підвищує вологість повітря, викликає корозію металевих виробів.

7. 6.2 Забруднення навколишнього середовища в магазині автозапчастин

В даній роботі досліджується навколишнє середовище (НС) магазину з продажу автозапчастин (Baza Auto).

Основними елементами, що забруднюють навколишнє середовище магазину є:

- транспортні засоби;
- тверді битові відходи;
- люмінісцентні лампи;
- металеві та паперові відходи.

Роздивимось забруднення навколишнього середовища за транспортними засобами.

У зв'язку з глобальною автоматизацією та механізацією, на дорогах міста з'являється все більше транспортних засобів. Більшість магазинів

розташовані безпосередньо біля найважливіших вулиць міста – пр. Леніна та пр. Героїв Сталінграду. Певна кількість клієнтів та співробітників мають персональний транспортний засіб.

Все це викликає, насамперед, забруднення повітря і підвищення в його складі концентрації шкідливих речовин.

Перетворення хімічної енергії палива в роботу пов'язане з утворенням токсичних і шкідливих речовин, які викидаються з відпрацьованих газів двигунів, продуктів зношування автомобільних шин, а також споживання в великих об'ємах палива і масла.

При цьому характерний вміст в повітрі таких шкідливих речовин, як CO, CH та NOx

Моноокис вуглецю CO страшна тим, що швидко попадає в кров, де засвоюється червоними кров'яними тельцями й утворює карбоксигемоглобін, що блокує здатність крові постачати клітини необхідним для життя киснем. Особливо чутливі до вдихання CO серцеві хворі й люди похилого віку - вони можуть відчувати біль у грудині, наприклад, піднімаючись по сходам. Але й ті, хто поки що здоровий, відчувають наслідки постійної інтоксикації - знижується увага, виникає почуття постійної втоми, пригніченості.

Тим часом у низько навислі над асфальтом хмарах CH і NOx під впливом світла відбуваються хімічні реакції. Розкладання окислів азоту приводить до утворення озону (O₃). Взагалі ж озон нестійкий і швидко розпадається, але тільки не в присутності вуглеводнів (CH) - вони сповільнюють процес розпаду озону, і він активно вступає в реакції із часточками вологи й інших з'єднань. Утворюється стійка хмара мутного смогу.

Крім цього, деякі вуглеводні CH є найсильнішими канцерогенними речовинами, озон роз'їдає очі й легені, а викиди NOx беруть участь у формуванні кислотних дощів.

Чадний газ і окисли азоту, що виділяються глушителем автомобілю, виступають причинами головних болів, втоми, невмотивованого роздратування, низької працездатності. Сірчистий газ впливає на свята

святих - генетичний апарат, сприяючи безплідності й уродженим каліцтвам. Всі ці фактори ведуть до стресів, нервовим проявам, прагненню до самоти, байдужності до найближчих людей. “Внесок” автомобільного транспорту в атмосферу становить 90% по окисі вуглецю й 70% по окисі азоту. Автомобіль додає в ґрунт і повітря важкі метали, інші шкідливі речовини.

У результаті спалювання рідкого палива в повітря щорічно викидається, за різними оцінками, від 180 тис. до 260 тис. т свинцевих часток, що в 60-130 разів перевершує природне надходження свинцю в атмосферу при вулканічних виверженнях (2-3 тис. т/рік). При вдиханні міського повітря великі свинцеві аерозолі затримуються в бронхах і носоглотці, а ті, що мають діаметр менш 1 мк (їх приблизно 70- 80%), попадають у легені, а потім проникають у капіляри й, з'єднуючись із еритроцитами, отруюють кров. Причому відомо, що “свинцеве повітря” шкідливіше “свинцевої води”. Ознаки свинцевого отруєння - анемія, постійні головні болі, м'язовий біль - проявляються при вмісті в крові свинцю 80 мкг/100 мл.

Аналізуючи ці дані, ми робимо висновок, що в процесі роботи транспортних засобів клієнтів та співробітників магазину здійснюється негативний вплив на навколишнє середовище, а саме, в атмосферу викидаються шкідливі речовини, які можуть призвести до різноманітних захворювань у людей. Вони також шкідливі і для рослин та забруднюють ґрунт навколо магазину, так як рослини теж дуже чутливі до забруднення – воно їх пригнічує, а шкідливі речовини осідають в ґрунті, порушую його склад і життєдіяльність організмів, які в ньому проживають. Тому потрібні заходи для зменшення впливу шкідливих факторів.

8. 6.3 Заходи щодо запобігання забруднень навколишнього середовища

В роботі з оздоровлення навколишнього середовища, обмеження впливів шкідливих речовин на біосферу важливу роль грають служби контролю

стану природи, середовища проживання людей, локального і регіонального моніторингу навколишнього середовища. Ці служби, озброєні сучасною вимірювальною технікою і приладами контролю, повинні оперативно оповіщати населення про всі випадки наближення параметрів навколишнього середовища до небезпечного рівня.

Обов'язок захищати навколишнє середовище закріплена в Конституції. Тому до компетенції центральних органів влади в області охорони атмосферного повітря віднесені:

- встановлення обмежень на викид забруднюючих речовин в повітря;
- визначення порядку державного обліку викидів забруднюючих речовин;
- нормування якості атмосферного повітря;
- визначення порядку видачі спеціальних дозволів (ліцензій) на викиди забруднюючих речовин;
- здійснення державного контролю за охороною атмосферного повітря.

Здійснення заходів щодо охорони атмосферного повітря не повинно приводити до забруднення інших об'єктів природного середовища.

Заходи щодо зниження забруднення НС транспортними засобами:

- регулярне переглядання в бік жорсткіших норм токсичності одиночними транспортними засобами;
- модернізація систем ДВЗ;
- підвищення якості моторного палива;
- нейтралізація відпрацьованих газів, фільтрація твердих часток.

7 Розрахунок економічного ефекту від впровадження підсистеми алгоритмів прогнозування даних

7.1 Вступна частина

На даний момент сучасні обчислювальні засоби широко використовуються в повсякденній діяльності людини. Сьогодні обчислювальна техніка дозволяє вирішувати найрізноманітніші задачі: від звільнення людини від монотонної ручної праці до розробки складних технічних розрахунків, що не під силу виконати одній людині. Сучасні засоби програмування дозволяють вирішувати задачі, які стояли в минулому перед цілими колективами наукових співробітників.

Розвиток нових технологій і ріст застосування обчислювальної техніки привели до широкого застосування ЕОМ для вирішення проблем, пов'язаних з обробкою інформації, її контролем і аналізом.

Призначенням розроблюваної системи є спрощення аналітичного віділу магазину з продажу автозапчастин.

Впровадження системи приведе до значної економії, що виражається в наступних показниках: зменшення витрат на закупівлю та зберігання зайвого товару; прогнозування закупівель товару що необхідний для продажу в необхідний час.

Найбільш важливим моментом для розроблювача, з економічної точки зору, є процес формування вартості системи. Очевидно, що вона являє собою дуже специфічний товар з безліччю особливостей.

Створення автоматизованої системи вимагає одноразових витрат на її розробку, придбання необхідних технічних засобів і поточних витрат на функціонування системи. Економія від функціонування автоматизованої системи визначається з урахуванням витрат на її експлуатацію. Відношення цієї економії до витрат на створення автоматизованої системи характеризує економічну ефективність капітальних вкладень. Економічні показники визначаються по діючим на момент розрахунку оптовим цінам, тарифам і ставкам заробітної плати.

7.2 Розрахунок вартості створення підсистеми магазину з продажу автозаачастин

Загальні витрати (на проектування, реалізацію, впровадження в експлуатацію програмного забезпечення "Автоматизоване робоче місце адміністратора магазину")

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

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

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

$C_{пз}$ – витрати на придбання та установку ПЗ, необхідного для розробки;

$C_{впр}$ – витрати на впровадження підсистеми (навчання персоналу і т.п.).

Витрати на проектування та реалізацію розраховуються за наступною формулою:

$$C_{п-р} = C_з + C_{від} + C_м + C_н,$$

де $C_з$ – заробітна плата розробників; $C_{від}$ – відрахування на соціальні заходи (відповідно до чинного законодавства $C_{від}$ становить 37,5% від фонду оплати праці); $C_м$ – вартість машинного часу, необхідного для розробки і налагодження; $C_н$ – накладні витрати у розмірі 50% 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 ПК).

На розробку данної програми необхідно 5 місяців (за експертною оцінкою часу на розробку аналогічних систем). Заробітна плата розроблювача за місяць дорівнює 850 грн.

$$C_з = 1800 * 5 = 9000 \text{ грн.}$$

$$C_{від} = 9000 * 0,375 = 3375 \text{ грн.}$$

Вартість машинного часу, необхідного для розробки і налагодження, обчислюється за формулою

$$C_M = t_m * C_{M-Г},$$

де t_m – тривалість машинного часу, необхідного для розробки, год.;

$C_{M-Г}$ – собівартість 1 машино-години роботи ПК.

$$C_{M-Г} = \frac{\sum_i U_i}{T_{ЕФ}},$$

де U_i - річні експлуатаційні витрати при обслуговуванні ПК, що охоплюють:

- заробітну плату спеціаліста, що обслуговує машину (з поправкою на коефіцієнт зайнятості робітника обслуговуванням ПК);
- нарахування на заробітну плату (37,5%);
- витрати на електроенергію, які визначаються як добуток вартості 1 кВт за годину електроенергії на обсяг потужності, що споживається ПК, і на ефективний фонд часу ПК (за рік);

- накладні витрати – 50% витрат на оплату праці;

$T_{ЕФ}$ – ефективний фонд часу роботи ПК за рік, що визначається виходячи з календарного річного фонду часу за винятком вихідних та святкових днів.

$$\text{Кількість робочих годин на рік} = 12 * 22 * 8 = 2112$$

$$C_{M-Г} = (800 + 800 * 0,375 + 800 * 0,5 + 250) / 2112 = 0,83 \text{ грн/год.}$$

$$C_M = 0,83 * 420 = 348,6 \text{ грн};$$

$$C_H = 9000 * 0,5 = 4500 \text{ грн};$$

$$C_{п-р} = 9000 + 3375 + 348,6 + 4500 = 17223,6 \text{ грн.}$$

Витрати на модернізацію устаткування для системи враховують вартість модернізації клієнта. (Згідно з цінами, які наводяться в прайс-листах фірми “ЛОГІКА”, витрати на модернізацію устаткування для системи дорівнює 1500 грн).

Витрати на придбання програмного забезпечення дорівнюють:

- 1) витрати на придбання середовища розробки - 1600 грн.
- 2) витрати на придбання мережної версії компонентів для розробки – 1000 грн.

У такий спосіб витрати на проектування, реалізацію, впровадження програмного забезпечення

$$З = C_{п-р} + C_{тз} + C_{пз} + C_{впр} = 17223,6 + 1500 + 1600 + 1000 = 21323,6 \text{ грн.}$$

Витрати на створення й експлуатацію програми $З_{се}$ в перший рік експлуатації обчислюється по наступній формулі:

$$З_{се} = З + З_{ер},$$

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),

$$A_{об} = 1500 * 0,6 = 900 \text{ грн.}$$

В масштабі магазина річні витрати на основні і допоміжні матеріали (гнучкі диски, папір) визначаються в розмірі 5% вартості основного устаткування.

$$З_{м} = 1500 * 0,05 = 75 \text{ грн.}$$

Річний обсяг робіт ПК у нормо-годинах визначається в такий спосіб:

$$\Phi_{м} = 253,3 * T_{с},$$

де $T_{с}$ – середнє місячне завантаження устаткування (близько 5 годин);

253,3 – середня кількість робочих днів у році.

$$\Phi_{м} = 253,3 * 5 = 1266,5 \text{ годин.}$$

Витрати на електроенергію $З_{е}$ при 1266,5 годиннах роботи устаткування в рік складуть (за умовою, що потужність ЕОМ складає 0,4 квт/год, а вартість 1 квт електроенергії складає 0,308 грн.):

$$З_{е} = 1266,5 * 0,308 * 0,4 = 156,03 \text{ грн.}$$

Експлуатаційні витрати на технічне забезпечення за рік складуть:

$$З_{ер} = 900 + 75 + 156,03 = 1131,03 \text{ грн.}$$

Отже, у перший рік витрати на створення й експлуатацію складуть:

$$З_{се} = 21323,6 + 1131,03 = 22454,63 \text{ грн.}$$

7.3 Розрахунок економічної ефективності

Основним показником економічної ефективності функціонування програми є підвищення ефективності керування у виді приросту прибутку, зниження витрат на керування при одночасному збільшенні швидкості і якості одержання потрібного результату.

До числа основних факторів, що визначають приріст прибутку в зв'язку з упровадженням програми, відносяться:

- збільшення швидкості одержання вхідної інформації;
- збільшення якості обробки вхідної інформації з одержанням достовірних результатів обробки даних;
- одержання необхідної для управлінської роботи інформації, зв'язаної з рішенням поставлених задач;
- підвищення продуктивності праці робітників, зниження імовірності виникнення помилок.

Крім того, не піддається прямій грошовій оцінці підвищення оперативності керування, якості прийнятих рішень, поліпшення організації праці і т.д.

Обов'язковою умовою визначення економічної ефективності програми є порівнянність усіх показників у часі, за цінами й іншими нормами, використовуваним для визначення показників, по змісту і колу елементів витрат.

За даними, які були отримані шляхом експериментальної оцінки, було встановлено, що використання данної системи дозволить вивільнити 10% робочого часу адміністратора магазину, що займається розподіленням навантаження на аналітичний відділ. Оскільки на даний момент працює 8 аналітиків, то, при впровадженні системи, аналітичний відділ можна спростити до одного працівника.

Визначимо пряму економічну ефективність, ґрунтуючись на тому, що автоматизація розподілення навантаження умовно вивільняє 7 .

Річна заробітна платня вивільнених працівників складає:

$$\Delta C_n = 7 * 1000 * 12 = 84000 \text{ грн.}$$

Річний економічний ефект розраховується за наступною формулою:

$$\mathcal{E}_{год} = \Delta C_n - E_n \cdot k$$

де E_n – нормативний коефіцієнт економічної ефективності капітальних вкладень у галузь – для обчислювальної техніки приймається – 0,6;

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

$$\mathcal{E}_{год} = 84000 - 0,6 * 22454,63 = 70527,22 \text{ грн}$$

Строк окупності системи розраховується за наступною формулою:

$$T = \frac{k}{\Delta C} = \frac{70527,22}{84000} = 0,8 \text{ (року)}$$

Таким чином, витрати на створення підсистеми розподілення навантаження окупляться протягом 0,8 року.

7.4 Висновки з економічної ефективності

В результаті розрахунків витрати на впровадження підсистеми алгоритма прогнозування даних (проектування і реалізація) складуть 21323,6 грн; експлуатаційні витрати на підсистему управління складуть 1131,03 грн. Загалом в перший рік витрати на модернізацію й експлуатацію складуть 22454,63 грн. Річна заробітня платня вивільнених працівників складає 84000 грн.

Витрати на розробку та впровадження підсистеми алгоритма прогнозування даних окупляться протягом 0,8 року.

За отриманими значеннями річного економічного ефекту та строку окупності можна зробити висновок, що впровадження підсистеми алгоритма прогнозування даних економічно вигідна і обґрунтована.

Висновки

В результаті виконання дипломної роботи була розроблена підсистема “Адміністратор магазинів”. Для цього були виконані аналіз предметної області та постановка задачі, розроблено технічне завдання та поетапні рішення автоматизованої системи, які включають:

- Загальносистемні рішення
- Рішення з інформаційного забезпечення
- Рішення з технічного забезпечення
- Рішення з програмного забезпечення

Крім того було виконано:

- Економічний розрахунок ефективності проекту
- Дослідження з охорони праці та охорони навколишнього середовища.

Розроблений програмний продукт вирішує проблему, яка пов’язана з неналежним рівнем автоматизації при здійсненні процесу замовлення товару, торгівлі автозапчастинами та формуванням звітів для аналізу ефективності магазину та бухгалтерського обліку.

З урахуванням вищенаведеного можна сказати, що цілі, поставлені задачу даної магістерської роботи були повністю виконані.

Список використаних джерел

1. Кириллов В.В. Основы проектирования реляционных баз данных. Учебное пособие. - СПб.: ИТМО, 1994. - 90 с.
2. ГОСТ 19.201-78. ГССС. Технічне завдання. Вимоги до змісту та оформлення.
3. ГОСТ 19.301-79. ГССС. Програма і методика випробувань. Вимоги до змісту та оформлення.
4. Мейер М. Теория реляционных баз данных. -М.: Мир, 1987. - 608 с.
5. Охорона навколишнього середовища. За редакцією Белова С.В. – М.: Вища школа, 1983. – 195 с.
6. Самгін Э.Б. Освітлення робочих місць. – М.: МИРЭА, 1989. – 186с.
7. Сибаров Ю.Г. «Охрана труда в вычислительных центрах». - Машиностроение, 1990. - 230 с.

Горбіков А.Ф. Використання алгоритмів прогнозування даних для магазину автозапчастин – Миколаїв : IX Всеукраїнська науково-практична конференція "Інформаційні технології в моделюванні", 2019.

1 Підстави для розробки

Документ “Про затвердження тем магістерських робіт”

2 Призначення розробки

Підсистема “Адміністратор магазинів” для удосконалення роботи магазину

3 Вимоги до програмного продукту

3.1 Вимоги до функціональних характеристик

Підсистема інтеграції повинна виконувати такі функції:

- дозволяти оперативно і точно задовольняти потреби покупця
- у режимі реального часу регулювати попит і потребу в товарі
- вирішувати складські та логістичні завдання
- формувати звіти по товарообігу
- вивід друку звіту, або накладної
- додавання необхідної інформації в базу даних
- надавання даних для бухгалтерського обліку

3.2 Вимоги до надійності

Потрібно слідкувати за введенням коректних даних. Деякі поля є необхідними для заповнення. Якщо користувач пропустить введення цієї інформації, програма має видати повідомлення, яке сповістить, що поля є необхідними для заповнення та запропонує ввести дані. Поки користувач не введе необхідні дані, програма функціонувати не буде. Це необхідно для того, щоб у подальшому програма функціонувала коректно. Доступ до програмного продукту повинні мати лише працівники магазину.

3.3 Вимоги до експлуатації

Даний програмний продукт не потребує особливих умов навколишнього середовища. Програма не потребує обслуговування. Від користувача вимагається знання інструкції користувача та вміння користуватися ПК.

3.4 Вимоги до інформаційної та програмної сумісності

1) “Адміністратор магазинів”

2) Операційна система Windows XP SP3/Vista/7/8/10

3) Віртуальна машина Java

3.5 Вимоги до маркування та пакування

Даний програмний продукт повинен розповсюджуватися на CD – дисках об'ємом 650-850 Мб.

3.6 Вимоги до транспортування та зберігання

Даний програмний продукт має ті ж вимоги транспортування та зберігання, що і CD – диски.

4 Вимоги до програмної документації

Програмна документація повинна містити:

- опис бази даних
- опис застосування бази даних
- програма
- інструкція користувача

5 Стадії та етапи розробки

Стадії та етапи розробки наведено в таблиці 1.

Таблиця 1 – основні етапи робіт за стадіями, терміни їх розробки та види документів

Стадія створення програмного проекту	Етапи робіт на стадії	Вид документу	Початок виконання	Кінець виконання
Перед проектне дослідження	Аналіз існуючих програмних продуктів, розробка пропонування про доцільність програмного продукту	Технічний звіт		
Технічні задачі	Розробка ТЗ, яке включає основні вимоги до програмного продукту	Технічне завдання		
Технічний проект	Вибір раціонального варіанту програмного продукту,	Технічний проект		

	задовольняючого вимогам технічного завдання			
Ескізний проект	Прийняття принципів рішень по структурі програмного продукту, усім видам забезпечення, попередній розрахунок економічної ефективності та технічних показників програмного продукту	Ескізний проект		
Робочий проект	Формування програмного продукту в цілому	Робочий проект		
Виготовлення, налагодження та іспит	Виготовлення та налагодження компонентів програмного продукту	Акт передачі програмного продукту		

6 Порядок прийому та контролю

Порядок контролю і здачі даного програмного продукту виконується представником замовника в присутності представника розробника згідно програми і методики випробування.

Для контролю і здачі даного програмного продукту повинна бути представлена інструкція користувача.

За результатами здачі складається акт, який підписується представником замовника і представником розробника, і затверджується керівниками організації-замовника і організації-розробника.

Розроблений програмний продукт має звичний графічний інтерфейс користувача, інтуїтивного розуміння, що виключить потребу спеціального навчання персоналу.

Одночасно з відкриттям програми з'являється вікно авторизації, яке пропонує ввести логін та пароль, або вийти із програми(рисунок 1).

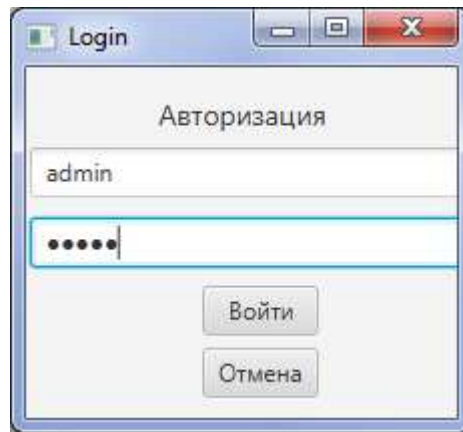


Рисунок 1 – Вікно авторизації

Якщо було введено коректний логін та пароль, програма автоматично перейде до головного меню (рисунок 2).

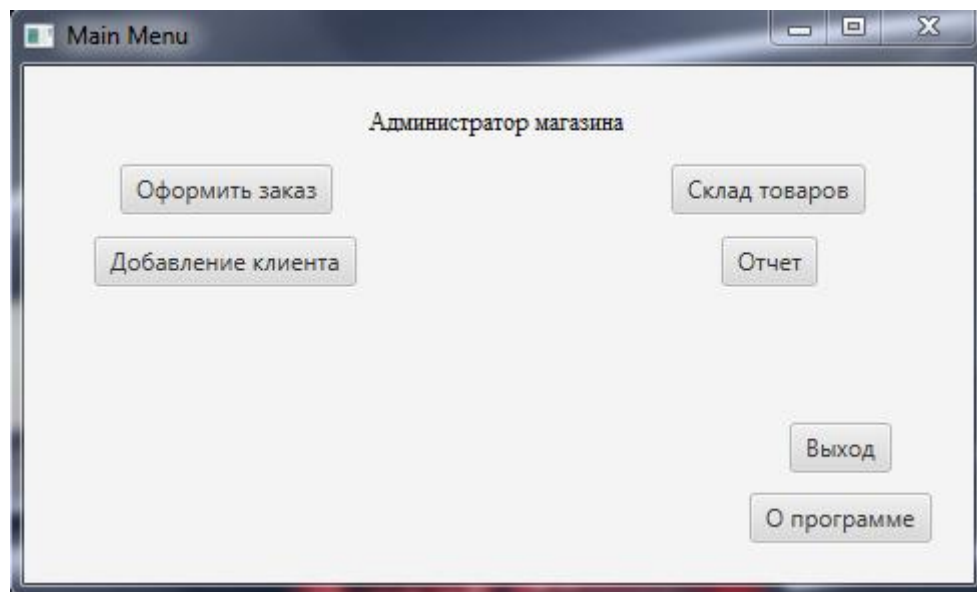


Рисунок 2 – Головне меню

Натиснувши кнопку головного меню “Оформить заказ”, відкривається вікно за допомогою якого можна сформувати заказ для покупця, та надрукувати його (рисунок 3).

The screenshot shows a software window titled "Ordering" with a subtitle "Оформление заказа". At the top right is a button "Печать накладной". Below the title, there are two input fields: "Покупатель:" with a dropdown menu showing "Иванов" and "Дата:" with a text box showing "29.05.2017" and a calendar icon. The main area contains a table with five columns: "Автомобиль", "Товар", "Количество", "Цена", and "Сумма". The first row of the table has the following values: "Opel,Astra,1999,1.6 16" in the "Автомобиль" column, "Амортизатор передн" in the "Товар" column, "2" in the "Количество" column, "255,55" in the "Цена" column, and "511,1" in the "Сумма" column. Below the table, there is a label "Всего к оплате:" and a button "Отмена".

Автомобиль	Товар	Количество	Цена	Сумма
Opel,Astra,1999,1.6 16	Амортизатор передн	2	255,55	511,1

Рисунок 3 – Вікно оформлення заказу

Натиснувши кнопку головного меню “Добавление клиента” відкривається вікно за допомогою якого, можна додати нового клієнта, якщо його ще не має в базі.(рисунок 4).

The screenshot shows a software window titled "New client" with a subtitle "Добавление клиента". It contains four input fields: "Имя" with the value "Василий", "Фамилия" with the value "Иванов", "Отчество" with the value "Иванович", and "Номер телефона" with the value "+380951234567". Below these fields are two buttons: "Добавить" and "Отмена".

Имя	Василий
Фамилия	Иванов
Отчество	Иванович
Номер телефона	+380951234567

Рисунок 4— Вікно додавання нового клієнта

Натиснувши кнопку головного меню “Склад товаров” відкривається база даних всіх товарів, які є на складі, та тих що потрібно замовити.(рисунок

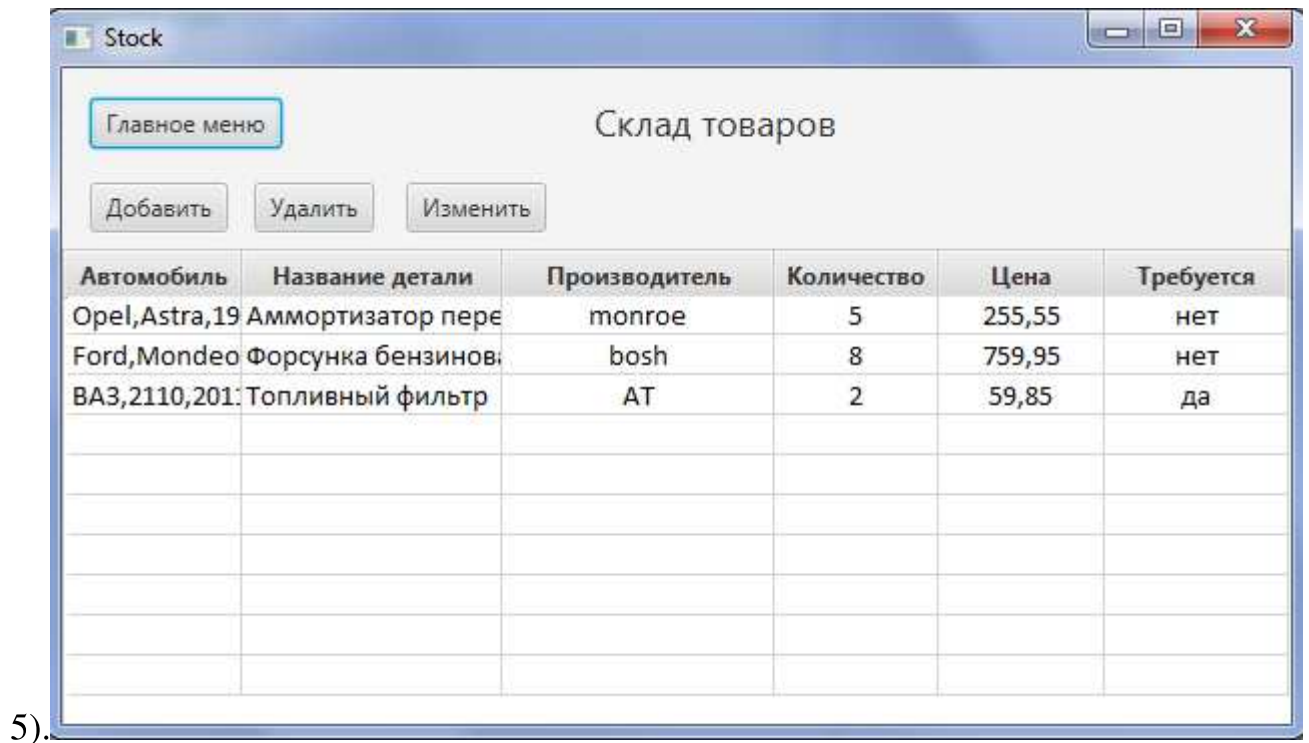


Рисунок 5 – Склад товаров

Якщо в вікні “Склад товаров” натиснути кнопку “Добавить” відкриється вікно, за допомогою якого можна додати новий товар на склад, або якщо обрати товар зі списку, та натиснути кнопку “Изменить”, то відкриється форма редагування товару (рисунок 6).

Добавление товара

Марка	Opel	Название детали	Амортизатор передний
Модель	Astra	Производитель	morgoe
Кузов	Sedan	Количество	5
Год	1999	Цена	255,55
Тип двигателя	1.6 16v benzin		

Добавить автомобиль

Добавить

Отмена

Рисунок 7 - Вікно додавання та редагування товару

В вікні “Добавление товара”, можна натиснути кнопку “Добавить автомобиль”, після чого відкривається вікно додавання нового автомобіля в базу (рисунок 8).

Добавление автомобиля

Марка	Opel
Модель	Astra
Кузов	Sedan
Год	1999
Тип двигателя	1.6 16v benzin

Добавить

Отмена

Рисунок 8 – Додавання нового автомобіля в базу

Натиснувши кнопку головного меню “Отчет” відкриється вікно формування звіту(рисунок 9).

The screenshot shows a window titled "Report" with a standard Windows-style title bar. The main content area is titled "Отчет". It features two date input fields: "С" (From) with the date "28.05.2017" and "По" (To) with the date "29.05.2017". To the right of these is a button labeled "За весь период". Below the date fields are three dropdown menus: "Покупатель" (Buyer) with the selected value "Иванов", "Автомобиль" (Car), and "Товар" (Goods). At the bottom of the window are three buttons: "Сформировать отчет" (Generate report), "Печать отчета" (Print report), and "Отмена" (Cancel). The "Сформировать отчет" button is highlighted with a blue border.

Рисунок 9 – Формування звіту

Розроблений і реалізований проект програмного забезпечення в середовищі IntelliJ IDEA 2016.2.5 на програмній мові Java.

Текст програми представлений в лістингах 1-21

Лістинг 1 – MainController.java

```
package main;

import javafx.collections.FXCollections;
import javafx.collections.ObservableList;
import javafx.event.EventHandler;
import javafx.fxml.FXML;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.control.TableColumn;
import javafx.scene.control.TableView;
import javafx.scene.control.cell.PropertyValueFactory;
import javafx.scene.input.MouseEvent;
import javafx.scene.layout.Pane;
import javafx.stage.Stage;
import tables.Parts;

import java.io.IOException;
import java.sql.SQLException;
import java.util.ArrayList;

public class Controller {

    @FXML
    private Button buttonEnter;
    @FXML
    private Button buttonExit;
    public static Stage stage;

    private ObservableList<Parts> partsData =
FXCollections.observableArrayList();

    @FXML
    private TableView<Parts> tableparts;

    @FXML
    private TableColumn<Parts, Integer> idCar;
    @FXML
    private TableColumn<Parts, Integer> count;

    @FXML
    private TableColumn<Parts, String> name;
```



```

@FXML
private TableColumn<Parts, String> manufacture;

@FXML
private TableColumn<Parts, String> need;

@FXML
private TableColumn<Parts, Double> price;

@FXML
public void initialize() {
    buttonEnter.addEventHandler(MouseEvent.MOUSE_CLICKED, new
EventHandler<MouseEvent>() {
        @Override
        public void handle(MouseEvent mouseEvent) {
            if (checkLogin()) {

                Parent root = null;
                try {
                    root =
FXMLLoader.load(getClass().getResource("../fxml/stock.fxml"));
                    stage.setTitle("main Menu");
                    stage.setScene(new Scene(root, 215, 175));
                    stage.show();
                } catch (IOException e) {
                    e.printStackTrace();
                }

            }
        }
    });
    buttonExit.addEventHandler(MouseEvent.MOUSE_CLICKED, new
EventHandler<MouseEvent>() {
        @Override
        public void handle(MouseEvent mouseEvent) {
            System.exit(0);
        }
    });

    count.setCellValueFactory(new PropertyValueFactory<Parts,
Integer>("count"));
    idCar.setCellValueFactory(new PropertyValueFactory<Parts,
Integer>("idCar"));
    name.setCellValueFactory(new PropertyValueFactory<Parts,
String>("name"));
    manufacture.setCellValueFactory(new
PropertyValueFactory<Parts, String>("manufacture"));
    need.setCellValueFactory(new PropertyValueFactory<Parts,
String>("need"));
    price.setCellValueFactory(new PropertyValueFactory<Parts,
Double>("price"));
    initData();
}

```

```

        // заполняем таблицу данными
        tableparts.setItems(partsData);
    }

    private void initData() {
        BaseLoad baseLoad = new BaseLoad();
        baseLoad.baseStart();
        try {
            ArrayList<Parts> parts =
baseLoad.selectParts(baseLoad.conn);
            partsData.addAll(parts);
        } catch (SQLException e) {

        }
    }

    private boolean checkLogin() {
        return true;
    }

    public void load(String pFileName, Stage pStage, String pTitle)
    {
        String fName = pFileName;

        Pane page = null;

        FXXMLLoader fxmlloader = new
FXXMLLoader(getClass().getResource(fName));

        try {
            page = (Pane) fxmlloader.load();
        } catch (IOException exception) {

        }

        Scene scene = new Scene(page);
        pStage.setScene(scene);
        pStage.setTitle(pTitle);
        pStage.show();
    }

}

```

Лістинг 2 – BaseLoad.java

```

package main;

import tables.Cars;
import tables.Parts;

import java.sql.*;
import java.util.ArrayList;

```

```

/**
 * Created by Arseniy on 16.06.2017.
 */
public class BaseLoad {

    String dbURL = "jdbc:mysql://localhost:3306/shop";
    String username = "root";
    String password = "root";
    public Connection conn;

    public void baseStart() {
        try {

            conn = DriverManager.getConnection(dbURL, username,
password);

            if (conn != null) {
                System.out.println("Connected");
            }
        } catch (SQLException ex) {
            ex.printStackTrace();
        }
    }

    public ArrayList<Cars> selectCars(Connection conn) throws
SQLException {
        String sql = "SELECT * FROM cars";
        ArrayList<Cars> parts = new ArrayList<>();

        Statement statement = conn.createStatement();
        ResultSet result = statement.executeQuery(sql);

        int count = 0;

        while (result.next()) {
            int id = Integer.parseInt(result.getString("id"));
            String model = result.getString("model");
            String year = result.getString("year");
            String make = result.getString("make");
            String engine = result.getString("engine");
            String mfrCode = result.getString("mfrCode");
            parts.add(new Cars(id, model, make, year, engine,
mfrCode));
        }

        return parts;
    }

    public ArrayList<Parts> selectParts(Connection conn) throws
SQLException {
        String sql = "SELECT * FROM parts";
        ArrayList<Parts> parts = new ArrayList<>();

```

```

        Statement statement = conn.createStatement();
        ResultSet result = statement.executeQuery(sql);

        int count = 0;

        while (result.next()) {
            int idCar =
Integer.parseInt(result.getString("idАвто"));
            int idPart =
Integer.parseInt(result.getString("idДетали"));
            int counts =
Integer.parseInt(result.getString("Количество"));
            String name = result.getString("Название детали");
            String manufacture = result.getString("Производитель");
            String need = result.getString("Требуется");
            Double price =
Double.parseDouble(result.getString("Цена"));
            parts.add(new Parts(idCar, idPart, counts, name,
manufacture, need, price));
        }

        return parts;
    }

    public void updatePrice(Connection conn, int id, double price)
throws SQLException {
        String sql = "UPDATE parts SET Цена=? WHERE idАвто=?";

        PreparedStatement statement = conn.prepareStatement(sql);
        statement.setString(1, Double.toString(price));
        statement.setString(2, Integer.toString(id));

        int rowsUpdated = statement.executeUpdate();
        if (rowsUpdated > 0) {
            System.out.println("Цена была успешно обновлена!");
        }
    }

    public void updateCount(Connection conn, int id, int count)
throws SQLException {
        String sql = "UPDATE parts SET Количество=? WHERE
idАвто=?";

        PreparedStatement statement = conn.prepareStatement(sql);
        statement.setString(1, Double.toString(count));
        statement.setString(2, Integer.toString(id));

        int rowsUpdated = statement.executeUpdate();
        if (rowsUpdated > 0) {
            System.out.println("Цена была успешно обновлена!");
        }
    }

```

```
    }  
}
```

Лістинг 3 – Main.java

```
package main;  
  
import javafx.application.Application;  
import javafx.fxml.FXMLLoader;  
import javafx.scene.Parent;  
import javafx.scene.Scene;  
import javafx.stage.Stage;  
  
public class Main extends Application {  
  
    public Parent root;  
  
    @Override  
    public void start(Stage primaryStage) throws Exception{  
        root =  
FXMLLoader.load(getClass().getResource("../fxml/authorization.fxml")  
);  
        primaryStage.setTitle("Authorization");  
        primaryStage.setScene(new Scene(root, 215, 175));  
        primaryStage.show();  
        Controller.stage = primaryStage;  
    }  
  
    public static void main(String[] args) {  
        launch(args);  
    }  
}
```

Лістинг 4 – Cars.java

```
package tables;  
  
import javax.persistence.Entity;  
import javax.persistence.GeneratedValue;  
import javax.persistence.GenerationType;  
import javax.persistence.Id;  
  
/**  
 * Created by Arseniy on 18.06.2017.  
 */  
@Entity  
public class Users {  
  
    @Id @GeneratedValue(strategy = GenerationType.IDENTITY)  
    private String id;  
  
    public String getUsername() {  
        return username;  
    }  
}
```

```

    }

    public void setUsername(String username) {
        this.username = username;
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;

        Users users = (Users) o;

        if (id != null ? !id.equals(users.id) : users.id != null)
return false;
        if (username != null ? !username.equals(users.username) :
users.username != null) return false;
        return password != null ? password.equals(users.password) :
users.password == null;

    }

    @Override
    public int hashCode() {
        int result = id != null ? id.hashCode() : 0;
        result = 31 * result + (username != null ?
username.hashCode() : 0);
        result = 31 * result + (password != null ?
password.hashCode() : 0);
        return result;
    }

    private String username;

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }

    private String password;

    public String getId() {
        return id;
    }

    public void setId(String id) {
        this.id = id;
    }

```

```
}
```

Лістинг 5 – Chklist.java

```
package tables;

import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;

/**
 * Created by Arseniy on 18.06.2017.
 */
@Entity
public class Chklist {
    @Id @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;

        Chklist chklist = (Chklist) o;

        if (id != chklist.id) return false;
        if (amount != chklist.amount) return false;
        return date == chklist.date;
    }

    @Override
    public int hashCode() {
        int result = id;
        result = 31 * result + amount;
        result = 31 * result + date;
        return result;
    }

    public int getAmount() {

        return amount;
    }

    public void setAmount(int amount) {
        this.amount = amount;
    }

    private int amount;
    private int date;
}
```

```

    public int getDate() {
        return date;
    }

    public void setDate(int date) {
        this.date = date;
    }

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }
}

```

Лістинг 6 – Parts.java

```

package tables;

/**
 * Created by Arseniy on 16.06.2017.
 */
public class Parts {

    private int idCar;
    private int idPart;
    private Integer count;
    private String name;
    private String manufacture;
    private String need;
    private double price;

    public Parts(int idCar, int idPart, int count, String name,
String manufacture, String need, double price) {
        this.idCar = idCar;
        this.idPart = idPart;
        this.count = count;
        this.name = name;
        this.manufacture = manufacture;
        this.need = need;
        this.price = price;
    }

    public Parts() {
    }

    public int getIdCar() {
        return idCar;
    }
}

```



```
public void setIdCar(int idCar) {
    this.idCar = idCar;
}

public int getIdPart() {
    return idPart;
}

public void setIdPart(int idPart) {
    this.idPart = idPart;
}

public Integer getCount() {
    return count;
}

public void setCount(Integer count) {
    this.count = count;
}

public String getName() {
    return name;
}

public void setName(String name) {
    this.name = name;
}

public String getManufacture() {
    return manufacture;
}

public void setManufacture(String manufacture) {
    this.manufacture = manufacture;
}

public String getNeed() {
    return need;
}

public void setNeed(String need) {
    this.need = need;
}

public double getPrice() {
    return price;
}

public void setPrice(double price) {
    this.price = price;
}
}
```

Лістинг 7 – Users.java

```
package tables;

import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;

/**
 * Created by Arseniy on 18.06.2017.
 */
@Entity
public class Users {

    @Id @GeneratedValue(strategy = GenerationType.IDENTITY)
    private String id;

    public String getUsername() {
        return username;
    }

    public void setUsername(String username) {
        this.username = username;
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;

        Users users = (Users) o;

        if (id != null ? !id.equals(users.id) : users.id != null)
            return false;
        if (username != null ? !username.equals(users.username) :
            users.username != null) return false;
        return password != null ? password.equals(users.password) :
            users.password == null;
    }

    @Override
    public int hashCode() {
        int result = id != null ? id.hashCode() : 0;
        result = 31 * result + (username != null ?
            username.hashCode() : 0);
        result = 31 * result + (password != null ?
            password.hashCode() : 0);
        return result;
    }

    private String username;

    public String getPassword() {
```

```

        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }

    private String password;

    public String getId() {
        return id;
    }

    public void setId(String id) {
        this.id = id;
    }
}

```

Лістинг 8 – Client.java

```

package tables;

import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;

/**
 * Created by Arseniy on 18.06.2017.
 */
@Entity
public class Client {
    @Id @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;

    public String getName() {
        return name;
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;

        Client client = (Client) o;

        if (id != client.id) return false;
        if (mobilenumber != client.mobilenumber) return false;
        if (!name.equals(client.name)) return false;
        if (!surname.equals(client.surname)) return false;
    }
}

```

```

        return patronimyc != null ?
patronimyc.equals(client.patronimyc) : client.patronimyc == null;

    }

    @Override
    public int hashCode() {
        int result = id;
        result = 31 * result + name.hashCode();
        result = 31 * result + surname.hashCode();
        result = 31 * result + (patronimyc != null ?
patronimyc.hashCode() : 0);
        result = 31 * result + mobilenumbr;
        return result;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getSurname() {
        return surname;
    }

    public void setSurname(String surname) {
        this.surname = surname;
    }

    public String getPatronimyc() {
        return patronimyc;
    }

    public void setPatronimyc(String patronimyc) {
        this.patronimyc = patronimyc;
    }

    public int getMobilenumbr() {
        return mobilenumbr;
    }

    public void setMobilenumbr(int mobilenumbr) {
        this.mobilenumbr = mobilenumbr;
    }

    private String name;
    private String surname;
    private String patronimyc;
    private int mobilenumbr;

    public int getId() {
        return id;
    }

    public void setId(int id) {

```

```

        this.id = id;
    }

}

```

Лістинг 9 – addauto.fxml

```

<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.text.*?>
<?import javafx.scene.control.*?>
<?import java.lang.*?>
<?import javafx.scene.layout.*?>
<?import javafx.geometry.Insets?>
<?import javafx.scene.layout.GridPane?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.control.Label?>

<AnchorPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity" prefHeight="331.0" prefWidth="318.0" xmlns="http://javafx.com/javafx/8" xmlns:fx="http://javafx.com/fxml/1">
    <children>
        <Label layoutX="56.0" layoutY="14.0" text="Добавление автомобиля">
            <font>
                <Font size="18.0" />
            </font>
        </Label>
        <Label layoutX="11.0" layoutY="106.0" text="Модель" />
        <Label layoutX="11.0" layoutY="68.0" text="Марка" />
        <Label layoutX="11.0" layoutY="144.0" text="Кузов" />
        <Label layoutX="11.0" layoutY="180.0" text="Год" />
        <Label layoutX="9.0" layoutY="213.0" text="Тип двигателя" />
        <Button layoutX="132.0" layoutY="264.0" mnemonicParsing="false" text="Добавить" />
        <Button layoutX="137.0" layoutY="296.0" mnemonicParsing="false" text="Отмена" />
        <TextField layoutX="112.0" layoutY="64.0" />
        <TextField layoutX="112.0" layoutY="102.0" />
        <TextField layoutX="112.0" layoutY="209.0" />
        <TextField layoutX="112.0" layoutY="176.0" />
        <TextField layoutX="112.0" layoutY="140.0" />
    </children>
</AnchorPane>

```

Лістинг 10 – addparts.fxml

```

<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.text.*?>
<?import javafx.scene.control.*?>

```

```

<?import java.lang.*?>
<?import javafx.scene.layout.*?>
<?import javafx.geometry.Insets?>
<?import javafx.scene.layout.GridPane?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.control.Label?>

<AnchorPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity" prefHeight="400.0" prefWidth="600.0"
xmlns="http://javafx.com/javafx/8"
xmlns:fx="http://javafx.com/fxml/1">
    <children>
        <Label layoutX="219.0" layoutY="14.0" text="Добавление
товара">
            <font>
                <Font size="18.0" />
            </font>
        </Label>
        <Label layoutX="322.0" layoutY="68.0" text="Название детали"
/>
        <Label layoutX="322.0" layoutY="106.0" text="Производитель"
/>
        <Label layoutX="322.0" layoutY="144.0" text="Количество" />
        <Label layoutX="322.0" layoutY="183.0" text="Цена" />
        <TextField layoutX="418.0" layoutY="64.0" />
        <TextField layoutX="418.0" layoutY="102.0" />
        <TextField layoutX="418.0" layoutY="140.0" />
        <TextField layoutX="418.0" layoutY="179.0" />
        <ChoiceBox layoutX="115.0" layoutY="64.0" prefWidth="150.0"
/>
        <ChoiceBox layoutX="115.0" layoutY="102.0" prefWidth="150.0"
/>
        <ChoiceBox layoutX="115.0" layoutY="140.0" prefWidth="150.0"
/>
        <ChoiceBox layoutX="115.0" layoutY="179.0" prefWidth="150.0"
/>
        <Label layoutX="11.0" layoutY="106.0" text="Модель" />
        <Label layoutX="11.0" layoutY="68.0" text="Марка" />
        <Label layoutX="11.0" layoutY="144.0" text="Кузов" />
        <Label layoutX="11.0" layoutY="183.0" text="Год" />
        <ChoiceBox layoutX="115.0" layoutY="221.0" prefWidth="150.0"
/>
        <Label layoutX="9.0" layoutY="225.0" text="Тип двигателя" />
        <Button layoutX="27.0" layoutY="269.0"
mnemonicParsing="false" text="Добавить автомобиль" />
        <Button layoutX="275.0" layoutY="314.0"
mnemonicParsing="false" text="Добавить" />
        <Button layoutX="281.0" layoutY="350.0"
mnemonicParsing="false" text="Отмена" />
    </children>
</AnchorPane>

```

Лістинг 11 – mainmenu.fxml

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.text.*?>
<?import javafx.scene.control.*?>
<?import java.lang.*?>
<?import javafx.scene.layout.*?>
<?import javafx.geometry.Insets?>
<?import javafx.scene.layout.GridPane?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.control.Label?>

<GridPane alignment="center" hgap="10" prefHeight="258.0"
prefWidth="422.0" vgap="10" xmlns="http://javafx.com/javafx/8"
        xmlns:fx="http://javafx.com/fxml/1"
fx:controller="main.Controller">
    <columnConstraints>
        <ColumnConstraints/>
    </columnConstraints>
    <rowConstraints>
        <RowConstraints/>
    </rowConstraints>
    <children>
        <AnchorPane minHeight="200.0" prefHeight="259.0"
prefWidth="432.0">
            <children>
                <Button layoutX="27.0" layoutY="49.0"
mnemonicParsing="false" text="Оформить заказ"/>
                <Button layoutX="302.0" layoutY="49.0"
mnemonicParsing="false" text="Склад товаров"/>
                <Button layoutX="327.0" layoutY="85.0"
mnemonicParsing="false" text="Отчет"/>
                <Button layoutX="361.0" layoutY="178.0"
mnemonicParsing="false" text="Выход"/>
                <Button layoutX="341.0" layoutY="213.0"
mnemonicParsing="false" text="О программе"/>
                <Button layoutX="14.0" layoutY="85.0"
mnemonicParsing="false" text="Добавление клиента"/>
                <Label layoutX="131.0" layoutY="14.0" text="APM
Админстратора">
                    <font>
                        <Font size="18.0"/>
                    </font>
                </Label>
            </children>
        </AnchorPane>
    </children>
</GridPane>
```

Лістинг 12 – newclient.fxml

```

<?xml version="1.0" encoding="UTF-8"?>

<?import java.lang.*?>
<?import javafx.scene.control.*?>
<?import javafx.scene.layout.*?>
<?import javafx.scene.text.*?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.control.PasswordField?>
<?import javafx.scene.control.TextField?>
<?import javafx.scene.layout.AnchorPane?>
<?import javafx.scene.layout.ColumnConstraints?>
<?import javafx.scene.layout.GridPane?>
<?import javafx.scene.layout.RowConstraints?>
<?import javafx.scene.text.Font?>

<VBox maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity" prefHeight="249.0" prefWidth="400.0"
xmlns="http://javafx.com/javafx/8"
xmlns:fx="http://javafx.com/fxml/1">
    <children>
        <AnchorPane prefHeight="218.0" prefWidth="400.0">
            <children>
                <Label layoutX="37.0" layoutY="64.0" text="Имя" />
                <Label layoutX="37.0" layoutY="101.0" text="Фамилия" />
                <Label layoutX="36.0" layoutY="137.0" text="Отчество" />
                <Label layoutX="35.0" layoutY="172.0" text="Номер
телефона" />
                <TextField layoutX="153.0" layoutY="133.0" />
                <TextField layoutX="153.0" layoutY="168.0" />
                <TextField layoutX="153.0" layoutY="60.0" />
                <TextField layoutX="153.0" layoutY="97.0" />
                <Button layoutX="166.0" layoutY="207.0"
mnemonicParsing="false" prefHeight="25.0" prefWidth="69.0"
text="Добавить" />
                <Label layoutX="115.0" layoutY="14.0" text="Добавление
клиента">
                    <font>
                        <Font size="18.0" />
                    </font>
                </Label>
            </children>
        </AnchorPane>
    </children>
</VBox>

```

Лістинг 13 – ordering.fxml

```

<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.geometry.*?>
<?import java.lang.*?>
<?import javafx.scene.control.*?>
<?import javafx.scene.layout.*?>
<?import javafx.scene.text.*?>

```



```

<?import javafx.scene.control.Button?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.control.PasswordField?>
<?import javafx.scene.control.TextField?>
<?import javafx.scene.layout.AnchorPane?>
<?import javafx.scene.layout.ColumnConstraints?>
<?import javafx.scene.layout.GridPane?>
<?import javafx.scene.layout.RowConstraints?>
<?import javafx.scene.text.Font?>

<VBox maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity" prefHeight="400.0" prefWidth="490.0"
xmlns="http://javafx.com/javafx/8"
xmlns:fx="http://javafx.com/fxml/1">
    <children>
        <AnchorPane prefHeight="97.0" prefWidth="490.0">
            <children>
                <Label layoutX="8.0" layoutY="53.0" text="Покупатель:">
                    <font>
                        <Font size="14.0" />
                    </font>
                </Label>
                <Label layoutX="162.0" layoutY="1.0" text="Оформление
заказа">
                    <font>
                        <Font size="18.0" />
                    </font>
                </Label>
                <ChoiceBox layoutX="96.0" layoutY="51.0"
prefWidth="150.0" />
                <Button layoutX="365.0" layoutY="24.0"
mnemonicParsing="false" text="Печать накладной" />
                <Label layoutX="269.0" layoutY="54.0" text="Дата">
                    <font>
                        <Font size="14.0" />
                    </font>
                </Label>
                <DatePicker layoutX="308.0" layoutY="53.0" />
            </children>
        </AnchorPane>
        <TableView prefHeight="200.0" prefWidth="200.0">
            <columns>
                <TableColumn prefWidth="87.0" text="Автомобиль" />
                <TableColumn prefWidth="125.0" text="Товар" />
                <TableColumn prefWidth="91.0" text="Количество" />
                <TableColumn prefWidth="77.0" text="Цена" />
                <TableColumn prefWidth="96.0" text="Сумма" />
            </columns>
        </TableView>
        <Label text="Всего к оплате:">
            <font>
                <Font size="18.0" />
            </font>
        </Label>
    </children>
</VBox>

```

```
    </children>
</VBox>
```

Лістинг 14 – report.fxml

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.text.*?>
<?import javafx.scene.control.*?>
<?import java.lang.*?>
<?import javafx.scene.layout.*?>
<?import javafx.geometry.Insets?>
<?import javafx.scene.layout.GridPane?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.control.Label?>

<AnchorPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity" prefHeight="400.0" prefWidth="600.0"
xmlns="http://javafx.com/javafx/8"
xmlns:fx="http://javafx.com/fxml/1">
    <children>
        <DatePicker layoutX="52.0" layoutY="49.0" />
        <DatePicker layoutX="300.0" layoutY="49.0" />
        <Label layoutX="20.0" layoutY="53.0" text="С" />
        <Label layoutX="270.0" layoutY="53.0" text="По" />
        <Button layoutX="499.0" layoutY="49.0"
mnemonicParsing="false" text="За весь период" />
        <Label layoutX="262.0" layoutY="14.0" text="Отчет">
            <font>
                <Font size="18.0" />
            </font>
        </Label>
        <Label layoutX="14.0" layoutY="106.0" text="Покупатель" />
        <ChoiceBox layoutX="93.0" layoutY="102.0" prefWidth="150.0"
/>
        <ChoiceBox layoutX="93.0" layoutY="144.0" prefWidth="150.0"
/>
        <Label layoutX="14.0" layoutY="148.0" text="Автомобиль" />
        <Label layoutX="12.0" layoutY="192.0" text="Товар" />
        <ChoiceBox layoutX="93.0" layoutY="188.0" prefWidth="150.0"
/>
        <Button layoutX="102.0" layoutY="242.0"
mnemonicParsing="false" text="Сформировать отчет" />
        <Button layoutX="214.0" layoutY="286.0"
mnemonicParsing="false" text="Отмена" />
        <Button layoutX="260.0" layoutY="242.0"
mnemonicParsing="false" text="Печать отчета" />
    </children>
</AnchorPane>
```

Лістинг 15 – authorization.fxml

```
<?xml version="1.0" encoding="UTF-8"?>
```

```

<?import javafx.scene.control.Button?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.control.PasswordField?>
<?import javafx.scene.control.TextField?>
<?import javafx.scene.layout.AnchorPane?>
<?import javafx.scene.layout.ColumnConstraints?>
<?import javafx.scene.layout.GridPane?>
<?import javafx.scene.layout.RowConstraints?>
<?import javafx.scene.text.Font?>

<GridPane alignment="center" hgap="10" vgap="10"
xmlns="http://javafx.com/javafx/8.0.112"
xmlns:fx="http://javafx.com/fxml/1"
fx:controller="main.Controller">
    <children>
        <AnchorPane prefHeight="140.0" prefWidth="227.0">
            <children>
                <Button fx:id="buttonExit" layoutX="94.0"
layoutY="130.0" mnemonicParsing="false" text="Отмена" />
                <Button fx:id="buttonEnter" layoutX="94.0"
layoutY="99.0" mnemonicParsing="false" prefHeight="25.0"
prefWidth="58.0" text="Войти" />
                <TextField layoutX="8.0" layoutY="30.0"
prefHeight="25.0" prefWidth="219.0" promptText="Username" />
                <Label layoutX="72.0" layoutY="4.0" text="Авторизация">
                    <font>
                        <Font size="14.0" />
                    </font>
                </Label>
                <PasswordField layoutX="8.0" layoutY="65.0"
prefHeight="25.0" prefWidth="219.0" promptText="Password" />
            </children>
        </AnchorPane>
    </children>
    <columnConstraints>
        <ColumnConstraints />
    </columnConstraints>
    <rowConstraints>
        <RowConstraints />
    </rowConstraints>
</GridPane>

```

Лістинг 16 – stock.fxml

```

<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.control.Button?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.control.TableColumn?>
<?import javafx.scene.control.TableView?>
<?import javafx.scene.layout.AnchorPane?>
<?import javafx.scene.text.Font?>

<AnchorPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-

```

```

Infinity" minWidth="-Infinity" prefHeight="453.0" prefWidth="617.0"
xmlns="http://javafx.com/javafx/8.0.112"
xmlns:fx="http://javafx.com/fxml/1"
fx:controller="main.Controller">
    <children>
        <Label layoutX="266.0" layoutY="14.0" text="Склад товаров">
            <font>
                <Font size="18.0" />
            </font>
        </Label>
        <Button layoutX="14.0" layoutY="57.0" mnemonicParsing="false"
text="Добавить" />
        <Button layoutX="96.0" layoutY="57.0" mnemonicParsing="false"
text="Удалить" />
        <Button layoutX="172.0" layoutY="57.0"
mnemonicParsing="false" text="Изменить" />
        <TableView fx:id="tableparts" layoutY="90.0">
            <columns>
                <TableColumn fx:id="idCar" prefWidth="89.0"
text="Автомобиль" />
                <TableColumn fx:id="name" prefWidth="130.0"
text="Название детали" />
                <TableColumn fx:id="manufacture" prefWidth="136.0"
text="Производитель" />
                <TableColumn fx:id="count" prefWidth="82.0"
text="Количество" />
                <TableColumn fx:id="price" prefWidth="83.0" text="Цена"
/>
                <TableColumn fx:id="need" prefWidth="92.0"
text="Требуется" />
            </columns>
        </TableView>
        <Button layoutX="14.0" layoutY="15.0" mnemonicParsing="false"
text="Главное меню" />
    </children>
</AnchorPane>

```

Лістинг 17 – CarsDAO.java

```

package dao;

import tables.Cars;

import java.sql.*;
import java.util.ArrayList;
import java.util.List;

public class CarsDAO {
    private Connection connection;

    public CarsDAO(Connection connection) {
        this.connection = connection;
    }
}

```

```

    public Cars create(String model, String make, int year, String
engine) {
        Cars cars = new Cars(model, make, year, engine);
        try {
            PreparedStatement query =
connection.prepareStatement("INSERT INTO cars (model, make, year,
engine) VALUES (?, ?, ?, ?)");
            query.setString(1,model);
            query.setString(2,make);
            query.setString(4,engine);
            query.setInt(3,year);
            query.executeUpdate();
            query.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return cars;
    }

    public List<Cars> findAll() {
        try {
            Statement query = connection.createStatement();
            ResultSet resultSet = query.executeQuery("SELECT * FROM
cars");

            List<Cars> result = new ArrayList<>();
            while (resultSet.next()) {
                String model = resultSet.getString("model");
                String make = resultSet.getString("make");
                String engine = resultSet.getString("engine");
                int year = resultSet.getInt("year");
                int id = resultSet.getInt("id");
                Cars cars = new Cars(model,make,year,engine);
                cars.setId(id);
                result.add(cars);
            }
            resultSet.close();
            query.close();
            return result;
        } catch (SQLException e) {
            return null;
        }
    }

    public void delete(Cars p) {
        try {
            PreparedStatement query =
connection.prepareStatement("DELETE FROM cars WHERE id = ?");
            query.setInt(1,p.getId());
            query.executeUpdate();
            query.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

```

```

    }

    public void delete(int id) {
        try {
            PreparedStatement query =
connection.prepareStatement("DELETE FROM cars WHERE id = ?");
            query.setInt(1,id);
            query.executeUpdate();
            query.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    public void update(Cars cars) {
        try {
            PreparedStatement query =
connection.prepareStatement("UPDATE cars SET model=?, make=?,
year=?, engine=? WHERE id = ?");
            query.setString(1, cars.getModel());
            query.setString(2, cars.getmake());
            query.setInt(3, cars.getyear());
            query.setString(4, cars.getengine());
            query.setInt(5, cars.getId());
            query.executeUpdate();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    public void create(Cars cars) {
        try {
            PreparedStatement query =
connection.prepareStatement("INSERT INTO cars (model, make, year,
engine) VALUES (?, ?, ?, ?)");
            query.setString(1, cars.getModel());
            query.setString(2, cars.getmake());
            query.setInt(3, cars.getyear());
            query.setString(4, cars.getengine());
            query.executeUpdate();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}

```

Лістинг 18 – ChklistDAO.java

```

package dao;

import tables.Chklist;

import java.sql.*;
import java.util.ArrayList;
import java.util.List;

```

```

public class ChklistDAO {
    private Connection connection;

    public ChklistDAO(Connection connection) {
        this.connection = connection;
    }

    public Chklist create(String amount, String date, int age,
String adress) {
        Chklist chklist = new Chklist(amount, date);
        try {
            PreparedStatement query =
connection.prepareStatement("INSERT INTO chklist (amount, date)
VALUES (?,?)");
            query.setString(1,amount);
            query.setString(2,date);
            query.executeUpdate();
            query.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return chklist;
    }

    public List<Chklist> findAll() {
        try {
            Statement query = connection.createStatement();
            ResultSet resultSet = query.executeQuery("SELECT * FROM
chklist");
            List<Chklist> result = new ArrayList<>();
            while (resultSet.next()) {
                String amount = resultSet.getString("amount");
                String date = resultSet.getString("date");
                int id = resultSet.getInt("id");
                Chklist chklist = new Chklist(amount,date);
                Chklist.setId(id);
                result.add(chklist);
            }
            resultSet.close();
            query.close();
            return result;
        } catch (SQLException e) {
            return null;
        }
    }

    public void delete(Chklist p) {
        try {
            PreparedStatement query =
connection.prepareStatement("DELETE FROM chklist WHERE id = ?");
            query.setInt(1,p.getId());
            query.executeUpdate();
            query.close();
        }
    }
}

```

```

        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    public void delete(int id) {
        try {
            PreparedStatement query =
connection.prepareStatement("DELETE FROM chklist WHERE id = ?");
            query.setInt(1, id);
            query.executeUpdate();
            query.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    public void update(Chklist Chklist) {
        try {
            PreparedStatement query =
connection.prepareStatement("UPDATE chklist SET amount=?, date=?
WHERE id = ?");
            query.setString(1, chklist.getamount());
            query.setString(2, chklist.getdate());
            query.setInt(5, chklist.getId());
            query.executeUpdate();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    public void create(Chklist chklist) {
        try {
            PreparedStatement query =
connection.prepareStatement("INSERT INTO chklist (amount, date)
VALUES (?,?)");
            query.setString(1, chklist.getamount());
            query.setString(2, chklist.getdate());
            query.executeUpdate();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}

```

Лістинг 19 – PartsDAO.java

```

package dao;

import tables.Parts;

import java.sql.*;
import java.util.ArrayList;
import java.util.List;

```



```

public class PartsDAO {
    private Connection connection;

    public PartsDAO(Connection connection) {
        this.connection = connection;
    }

    public Parts create(String count, String name, int price,
String manufacture) {
        Parts parts = new Parts(count, name, price, manufacture);
        try {
            PreparedStatement query =
connection.prepareStatement("INSERT INTO parts (count, name, price,
manufacture) VALUES (?, ?, ?, ?)");
            query.setString(1, count);
            query.setString(2, name);
            query.setString(4, manufacture);
            query.setInt(3, price);
            query.executeUpdate();
            query.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return parts;
    }

    public List<Parts> findAll() {
        try {
            Statement query = connection.createStatement();
            ResultSet resultSet = query.executeQuery("SELECT * FROM
parts");

            List<Parts> result = new ArrayList<>();
            while (resultSet.next()) {
                String count = resultSet.getString("count");
                String name = resultSet.getString("name");
                String manufacture =
resultSet.getString("manufacture");
                int price = resultSet.getInt("price");
                int id = resultSet.getInt("id");
                Parts parts = new
Parts(count, name, price, manufacture);
                parts.setId(id);
                result.add(parts);
            }
            resultSet.close();
            query.close();
            return result;
        } catch (SQLException e) {
            return null;
        }
    }
}

```

```

    public void delete(Parts p) {
        try {
            PreparedStatement query =
connection.prepareStatement("DELETE FROM parts WHERE id = ?");
            query.setInt(1,p.getId());
            query.executeUpdate();
            query.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    public void delete(int id) {
        try {
            PreparedStatement query =
connection.prepareStatement("DELETE FROM parts WHERE id = ?");
            query.setInt(1,id);
            query.executeUpdate();
            query.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    public void update(Parts parts) {
        try {
            PreparedStatement query =
connection.prepareStatement("UPDATE parts SET count=?, name=?,
price=?, manufacture=? WHERE id = ?");
            query.setString(1, parts.getCount());
            query.setString(2, parts.getName());
            query.setInt(3, parts.getPrice());
            query.setString(4, parts.getManufacture());
            query.setInt(5, parts.getId());
            query.executeUpdate();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    public void create(Parts parts) {
        try {
            PreparedStatement query =
connection.prepareStatement("INSERT INTO parts (count, name, price,
manufacture) VALUES (?, ?, ?, ?)");
            query.setString(1, parts.getCount());
            query.setString(2, parts.getName());
            query.setInt(3, parts.getPrice());
            query.setString(4, parts.getManufacture());
            query.executeUpdate();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

```

```
}  
}
```

Лістинг 20 – ClientDAO.java

```
package dao;  
  
import tables.Client;  
  
import java.sql.*;  
import java.util.ArrayList;  
import java.util.List;  
  
public class ClientDAO {  
    private Connection connection;  
  
    public ClientDAO(Connection connection) {  
        this.connection = connection;  
    }  
  
    public Client create(String name, String surname, int  
mobilenumber, String patronimic) {  
        Client client = new Client(name, surname, patronimic,  
mobilenumber);  
        try {  
            PreparedStatement query =  
connection.prepareStatement("INSERT INTO client (name, surname,  
patronimic, mobilenumber) VALUES (?, ?, ?, ?)");  
            query.setString(1, name);  
            query.setString(2, surname);  
            query.setString(4, patronimic);  
            query.setInt(3, mobilenumber);  
            query.executeUpdate();  
            query.close();  
        } catch (SQLException e) {  
            e.printStackTrace();  
        }  
        return client;  
    }  
  
    public List<Client> findAll() {  
        try {  
            Statement query = connection.createStatement();  
            ResultSet resultSet = query.executeQuery("SELECT * FROM  
client");  
            List<Client> result = new ArrayList<>();  
            while (resultSet.next()) {  
                String name = resultSet.getString("name");  
                String surname = resultSet.getString("surname");  
                String patronimic =  
resultSet.getString("patronimic");  
                int mobilenumber =
```

```

resultSet.getInt("mobilenumber");
        int id = resultSet.getInt("id");
        Client client = new
Client(name,surname,patronimic,patronimic);
        client.setId(id);
        result.add(client);
    }
    resultSet.close();
    query.close();
    return result;
} catch (SQLException e) {
    return null;
}
}

public void delete(Client p) {
    try {
        PreparedStatement query =
connection.prepareStatement("DELETE FROM client WHERE id = ?");
        query.setInt(1,p.getId());
        query.executeUpdate();
        query.close();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

public void delete(int id) {
    try {
        PreparedStatement query =
connection.prepareStatement("DELETE FROM client WHERE id = ?");
        query.setInt(1,id);
        query.executeUpdate();
        query.close();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

public void update(Client client) {
    try {
        PreparedStatement query =
connection.prepareStatement("UPDATE client SET name=?, surname=?,
patronimic=?, patronimic=? WHERE id = ?");
        query.setString(1, client.getName());
        query.setString(2, client.getSurname());
        query.setInt(3, client.getmobilenumber());
        query.setString(4, client.getpatronimic());
        query.setInt(5, client.getId());
        query.executeUpdate();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
}

```

```

        public void create(Client client) {
            try {
                PreparedStatement query =
connection.prepareStatement("INSERT INTO client (name, surname,
mobilenumber, patronimic) VALUES (?, ?, ?, ?)");
                query.setString(1, client.getname());
                query.setString(2, client.getsurname());
                query.setInt(3, client.getmobilenumber());
                query.setString(4, client.getpatronimic());
                query.executeUpdate();
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
    }
}

```

Лістинг 21- UsersDAO.java

```

package dao;

import tables.Users;

import java.sql.*;
import java.util.ArrayList;
import java.util.List;

public class UsersDAO {
    private Connection connection;

    public UsersDAO(Connection connection) {
        this.connection = connection;
    }

    public Users create(String username, String password) {
        Users Users = new Users(username, password);
        try {
            PreparedStatement query =
connection.prepareStatement("INSERT INTO Users (username, password)
VALUES (?, ?)");
            query.setString(1, username);
            query.setString(2, password);
            query.executeUpdate();
            query.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return Users;
    }

    public List<Users> findAll() {
        try {
            Statement query = connection.createStatement();

```

```

        ResultSet resultSet = query.executeQuery("SELECT * FROM
Users");
        List<Users> result = new ArrayList<>();
        while (resultSet.next()) {
            String username = resultSet.getString("username");
            String password = resultSet.getString("password");
            int id = resultSet.getInt("id");
            Users Users = new Users(username,password);
            Users.setId(id);
            result.add(Users);
        }
        resultSet.close();
        query.close();
        return result;
    } catch (SQLException e) {
        return null;
    }
}

public void delete(Users p) {
    try {
        PreparedStatement query =
connection.prepareStatement("DELETE FROM Users WHERE id = ?");
        query.setInt(1,p.getId());
        query.executeUpdate();
        query.close();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

public void delete(int id) {
    try {
        PreparedStatement query =
connection.prepareStatement("DELETE FROM Users WHERE id = ?");
        query.setInt(1,id);
        query.executeUpdate();
        query.close();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

public void update(Users Users) {
    try {
        PreparedStatement query =
connection.prepareStatement("UPDATE Users SET username=?,
password=?,WHERE id = ?");
        query.setString(1, Users.getUsername());
        query.setString(2, Users.getPassword());
        query.setInt(5, Users.getId());
        query.executeUpdate();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

```

```
    }  
}  
  
public void create(Users Users) {  
    try {  
        PreparedStatement query =  
connection.prepareStatement("INSERT INTO Users (username, password)  
VALUES (?,?)");  
        query.setString(1, Users.getUsername());  
        query.setString(2, Users.getPassword());  
        query.executeUpdate();  
    } catch (SQLException e) {  
        e.printStackTrace();  
    }  
}  
}
```