МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Чорноморський національний університет
імені Петра Могили
Факультет комп'ютерних наук
Кафедра інтелектуальних інформаційних систем

ДОПУЩЕНО ДО ЗАХИСТУ
Завідувач кафедри інтелектуальних
інформаційних систем, д.т.н., проф.
_____ Ю.П. Кондратенко
«____»_____2022 року

БАКАЛАВРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

ІНФОРМАЦІЙНА СИСТЕМА ПРИЙОМУ ЗАМОВЛЕНЬ У СЕРВІС-ЦЕНТРІ ПО ОБСЛУГОВУВАННЮ МОБІЛЬНИХ ПРИСТРОЇВ

Спеціальність 122 «Комп'ютерні науки»

122 – БКР – 401.21810110

Виконав студент 4-го курсу, групи 401
_____О. І. Желтобрюхов
«___» червня 2022 р.

Керівник канд. пед. наук, доцент
_____Н. М. Болюбаш
«___» червня 2022 р.

Миколаїв – 2022

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Чорноморський національний університет ім. Петра Могили
Факультет комп'ютерних наук
Кафедра інтелектуальних інформаційних систем

Рівень вищої освіти **бакалавр**
Спеціальність **122 «Комп'ютерні науки»**
(шифр і назва)
Галузь знань **12 «Інформаційні технології»**
(шифр і назва)

ЗАТВЕРДЖУЮ

Завідувач кафедри інтелектуальних інформаційних систем, д-р техн. наук, проф.
_____ Ю. П. Кондратенко
« » 20 р.

З А В Д А Н Н Я

на виконання кваліфікаційної роботи

Видано студенту групи 401 факультету комп'ютерних наук Желтобрюхову Олегу Ігоровичу.

1.Тема кваліфікаційної роботи «Інформаційна система прийому замовлень у сервіс-центрі по обслуговуванню мобільних пристроїв».

Керівник роботи Болюбаш Надія Миколаївна, канд. пед. наук, доцент.
Затв. наказом Ректора ЧНУ ім. Петра Могили від «__» ____ 202_ р. № _____

2. Строк представлення кваліфікаційної роботи студентом «__» _____ 2022 р.

3. Вхідні (початкові) дані до роботи: діяльність сервіс-центрів по обслуговуванню мобільних пристроїв, їх функціональні можливості, сукупність послуг.

Очікуваний результат: інформаційна система прийому замовлень у сервіс-центрі обслуговування мобільних пристроїв.

4. Перелік питань, що підлягають розробці (зміст пояснювальної записки):

— аналіз теоретичних засад діяльності центрів у сфері обслуговування мобільних пристроїв та виявлення сучасного стану програмного забезпечення по прийому та обробці замовлень у сервісних центрах;
— дослідження та обґрунтування вибору методу проєктування інформаційної системи;
— обґрунтування вибору інструментальних засобів розробки та моделювання автоматизованої інформаційної системи прийому замовлень у сервіс-центрі;
— здійснення програмної реалізації та тестування інформаційної системи прийому та обробки замовлень сервіс-центру з обслуговування мобільних пристроїв.

5. Перелік графічного матеріалу: презентація, __ графіків та __ додатків.

6. Завдання до спеціальної частини: санітарно-гігієнічні вимоги до робочого місця користувача ПК.

7. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис
Спеціальна частина з охорони праці	ст. викладач О.В. Макарова	

Керівник роботи __канд. пед. наук, доц. Болюбаш Н. М._____
(наук. ступінь, вчене звання, прізвище та ініціали)

_____
(підпис)

Завдання прийнято до виконання ___Желтобрюхов О. І._____
(прізвище та ініціали)

_____
(підпис)

Дата видачі завдання «_____» _____ 20__ р.

КАЛЕНДАРНИЙ ПЛАН
виконання кваліфікаційної роботи

Тема: «Інформаційна система прийому замовлень у сервіс-центрі по обслуговуванню мобільних пристроїв»

№	Найменування роботи	Початок	Закінчення	Примітки
1	Подання заяви на затвердження теми та керівника БКР	26.10.2021	30.10.2021	Виконано
2	Отримання завдання на виконання БКР	25.11.2021	25.11.2021	Виконано
3	Складання календарного плану роботи на весь період виконання БКР	26.112021	10.12.2021	Виконано
4	Огляд літературних джерел, аналіз предметної області, аналогів ПЗ у сфері обслуговування мобільних пристроїв	11.12.2021	31.12.2021	Виконано
5	Створення дизайну, проєктування та програмна реалізація інформаційної системи	1.01.2022	1.04.2022	Виконано
6	Робота над розділами пояснювальної записки БКР	16.04.2022	15.05.2022	Виконано
8	Розробка спеціальної частини з охорони праці	16.05.2022	22.05.2022	Виконано
9	Проходження переддипломної практики, збір та аналіз матеріалів, остаточне оформлення розділів БКР	26.05.2022	05.06.2022	Виконано
10	Попередній захист БКР	30.05.2022	30.05.2022	Виконано
11	Обговорення отриманих результатів з керівником, доробка та остаточне оформлення БКР	1.06.2022	16.05.2022	Виконано
12	Подання БКР рецензенту	16.05.2022	18.05.2022	Виконано
13	Створення слайдів для захисту та написання доповіді	18.06.2022	20.06.2022	Виконано
14	Подання БКР, її електронної копії та інших документів до захисту	20.06.2022	22.06.2022	Виконано
15	Захист БКР перед ЕК	27.06.2022		Виконано

Розробив студент_____Желтобрюхов О. І._____ _____
(прізвище та ініціали) *(підпис)*

Керівник __канд. пед. н, доцент Болюбаш Н.М_____ _____
(наук. ступінь, вчене звання, прізвище та ініціали) *(підпис)*

« » _____ 202_ р.

АНОТАЦІЯ
до бакалаврської роботи

Тема: «Інформаційна система прийому замовлень у сервіс-центрі по обслуговуванню мобільних пристроїв»

Студент: Желтобрюхов Олег Ігорович
Керівник: к.пед.н, доцент Болюбаш Надія Миколаївна

Бакалаврська кваліфікаційна робота присвячена проєктуванню, розробці та здійсненню програмної реалізації інформаційної системи прийому замовлень у сервіс-центрі по обслуговуванню мобільних пристроїв.

Об'єкт дослідження – сервісна технічна підтримка та обслуговування мобільних пристроїв.

Предмет дослідження – web-орієнтовані програмні засоби для обробки замовлень у сервіс-центрі по обслуговуванню мобільних пристроїв.

Метою роботи є поліпшення роботи сервісного центру по обслуговуванню мобільних пристроїв шляхом розробки і впровадження автоматизованої інформаційної системи прийому замовлень із можливістю відслідковування етапів ремонту у режимі реального часу.

Бакалаврська кваліфікаційна робота складається з фахової частини і спеціальної частини з охорони праці. Пояснювальна записка БКР складається зі вступу, чотирьох розділів, висновків та додатків.

У першому розділі розкрито теоретичні засади сфери обслуговування мобільних пристроїв та виявлено сучасний стан програмного забезпечення по прийому та обробці замовлень у сервісних центрах. У другому розділі здійснено огляд та обґрунтовано вибір методу проєктування ІС. У третьому розділі описано технології та інструментальні засоби розробки автоматизованої ІС, у четвертому – програмну реалізацію та тестування інформаційної системи. У четвертому розділі розкрито питання спеціальної частини з охорони праці.

Бакалаврська кваліфікаційна робота містить 83 сторінки, 21 рисунок, 22 джерела, 1 додаток.

ABSTRACT
for bachelor's work

Subject: «Information system for receiving orders in the service center for mobile devices»

Student: Zheltobriukhov Oleg Igorovich
Leader: Ph.D., associate professor Bolyubash Nadiya Mikolaivna

Bachelor's thesis is devoted to the design, development and implementation of software implementation of the information system for receiving orders in the service center for mobile devices.

The object of research is service technical support and maintenance of mobile devices.

The subject of the research is a web-oriented software for order processing in the service center for mobile devices.

The purpose of the thesis is to increase the efficiency of the service center for the maintenance of mobile devices by developing and implementing an automated information system for receiving orders with the ability to track the stages of repair in real time.

Thesis consists of a professional part and a special part on labor protection. Explanatory note of the thesis consists of an introduction, four chapters, conclusions and appendix.

The first section reveals the theoretical foundations of mobile devices and reveals the current state of software for receiving and processing orders in service centers. The second section reviews and justifies the choice of IS design method. The third section describes the technologies and tools for the development of automated IS, the fourth - software implementation and testing of information systems. The fourth chapter reveals the issue of a special part of labor protection.

Thesis contains 83 pages (without appendix), 21 figures, 22 sources, 1 supplements.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

ІС – інформаційна система

СУБД – система управління базами даних

DFD – Data Flow Diagrams

HTML – HyperText Markup Language

IDEF0 – Icam DEFinition

MVC – Model View Controller

MVVM – Model View ViewModel

SADT – Structured Analysis and Design Technique

Пояснювальна записка

до кваліфікаційної роботи

на тему:

ІНФОРМАЦІЙНА СИСТЕМА ПРИЙОМУ ЗАМОВЛЕНЬ У СЕРВІС-ЦЕНТРІ ПО ОБСЛУГОВУВАННЮ МОБІЛЬНОЇ ТЕХНІКИ

Спеціальність 122 «Комп’ютерні науки»

122 – БКР – 401.21810110

Виконав студент 4-го курсу, групи 401
_____О. І. Желтобрюхов
«___» червня 2022 р.

Керівник: канд. пед. наук, доцент
_____Н. М. Болюбаш
«___» червня 2022 р.

Миколаїв – 2022

ВСТУП

Актуальність. На даний момент у світовій спільноті склалася така ситуація, що інформаційні технології зайняли свою нішу у всіх сферах життєдіяльності суспільства. Це відбувається у зв'язку з глобальним збільшенням потоків інформації. Зручність та ефективність використання мережевих програмних засобів для підтримки діяльності у сфері обслуговування мобільних пристроїв є суттєвою та має тенденцію до їхньої постійної затребуваності. Застосування обліку в електронному вигляді дозволяє зекономити час, затрачений на обробку даних, їх облік та аналіз, це дозволяє підвищити ефективність управлінських рішень, а також забезпечує швидкий та своєчасний доступ до необхідної інформації.

Сьогодні у сфері сервісного обслуговування мобільних пристроїв існує чимало мережевих інтернет-ресурсів, які надають змогу ознайомитися із самим сервіс-центром, його контактами та послугами. Однак лише деякі з них дозволяють повноцінно надавати онлайн-підтримку технічного супроводу, відслідковуючи у режимі реального часу стан ремонту, вартість робіт тощо. Одним із шляхів вирішення даної проблеми є розробка та впровадження у діяльність сервіс-центрів інформаційної системи, яка забезпечує прийом та обробку заявок клієнтів у режимі реального часу.

Це обумовило **мету дослідження**, яка полягає у поліпшенні роботи сервісного центру по обслуговуванню мобільних пристроїв шляхом розробки і впровадження автоматизованої інформаційної системи прийому замовлень із можливістю відслідковування етапів ремонту у режимі реального часу.

Для досягнення поставленої мети було поставлено такі **завдання**:

– здійснити аналіз теоретичних засад діяльності центрів у сфері обслуговування мобільних пристроїв та виявити сучасний стан програмного забезпечення по прийому та обробці замовлень у сервісних центрах;

– дослідити та обґрунтувати вибір методу проєктування інформаційної системи;

– обґрунтувати вибір інструментальних засобів розробки та здійснити моделювання автоматизованої інформаційної системи прийому замовлень у сервіс-центрі;

– здійснити програмну реалізацію та тестування інформаційної системи прийому та обробки замовлень сервіс-центру з обслуговування мобільних пристроїв.

Об’єкт дослідження – сервісна технічна підтримка та обслуговування мобільних пристроїв.

Предмет дослідження – web-орієнтовані програмні засоби для обробки замовлень у сервіс-центрі по обслуговуванню мобільних пристроїв.

Методологічною основою дослідження є загальнонаукові та спеціальні методи, які дозволили вивчити предмет та об’єкт дослідження, дослідити напрями та шляхи оптимізації процесу подачі заявки на ремонт смартфонів та інших мобільних пристроїв.

Практичне значення отриманих результатів полягає в тому, що використання розробленого застосунку в діяльності сервіс-центру дозволить підвищити якість наданих послуг та допоможе полегшити процес оформлення заявки на ремонт та спілкування клієнтів зі співробітниками центру.

Структура роботи. Відповідно до мети, завдань і предмета дослідження, бакалаврська кваліфікаційна робота містить основну та спеціальну частини. Основна частина роботи складається зі вступу, чотирьох розділів, висновку, списку використаних джерел та одного додатку. Загальний обсяг роботи 83 сторінки, із них основного тексту основної частини 53 сторінок, спеціальної 16 сторінок. Кількість використаних джерел 22.

1 ТЕОРЕТИЧНІ ЗАСАДИ ДІЯЛЬНОСТІ ЦЕНТРІВ СЕРВІСНОГО ОБСЛУГОВУВАННЯ МОБІЛЬНИХ ПРИСТРОЇВ

1.1 Інформаційна система для підтримки діяльності сервісного центру

Сервісний центр – організація, що займається наданням послуг із сервісної підтримки та обслуговування техніки, обладнання та іншої продукції. Діяльність сервісних центрів включає перед торговий, гарантійний та після продажний ремонт, та постачання виробленої продукції. Сервісний центр як організація складається із п'яти відділів. Кожен відділ виконує свою функцію, завдяки чому на виході виходить кінцевий продукт організації - несправність у техніці усунута вчасно, у оголошений бюджет і нових не виявлено. Перелік відділів наведено нижче [1]:

1) Кадровий відділ - відбір та наймання персоналу. Також веде аналітику роботи організації.

2) Відділ комерції - відповідає за приплив грошей у сервісний центр, маркетинг, рекламу, спілкування з клієнтами, укладання угод.

3) Фінансовий відділ - відповідає за підрахунок грошей, здачу звітів до перевіряючих органів та контролю матеріальних активів.

4) Відділ виробництва - відповідає за виконання ремонтів та усунення несправностей у техніці. Сюди відноситься постачання, майстри, інженери, закупівля запчастин та інша логістика.

5) Контроль якості - відділ, який перевіряє техніку на усунення дефекту після ремонту. Також навчає персонал на предмет підвищення кваліфікації та ознайомлення з новими способами ремонтів.

Чим глобальніше підприємство у кількості робіт та за кількістю штатних одиниць, тим більше у ньому виробничих служб, відділів, інформаційних каналів, запропонованих інструкцій. Тому сильніше формалізовані та знеособлені

службові контакти, пов’язані з отриманням вказівок та здійсненням контролю. На рисунку 1.1 зображено бізнес-процес діяльності сервісного центру.

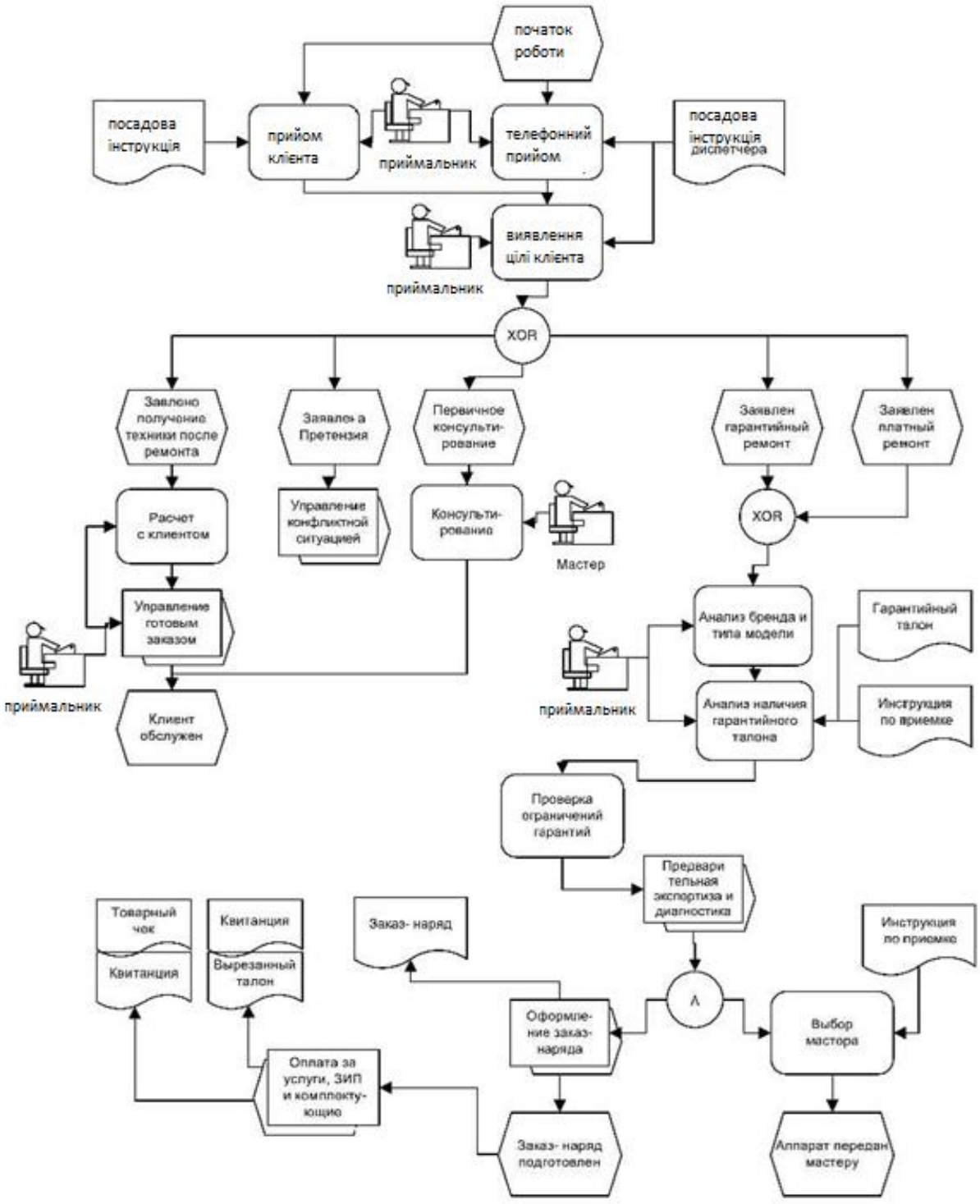

Рисунок 1.1 - Бізнес-процес діяльності сервісного центру

Інформаційна система - це сукупність програмних та апаратних засобів, а також організаційне забезпечення, які разом надають інформаційну підтримку людині в різних сферах її діяльності. Їхнє основне завдання полягає в тому, щоб збирати, зберігати, обробляти та передавати інформацію. Кожна певна інформаційна система спрямовано підтримку виконання операцій та забезпечення прийняття вірних рішень за умов поставленої нею завдання [2].

Одна з найважливіших ролей інформаційної системи – здатність отримувати необроблені дані та перетворювати їх на корисну інформацію, яку потім можна використовувати на підприємстві.

Для забезпечення ефективної роботи сервісного центру необхідно використовувати інформаційну систему, створену спеціально для цієї мети.

Під час виконання функцій управління чи будь-якого бізнес-процесу на підприємстві, чи то у сфері виробництва чи послуг, успішна діяльність буде залежати безпосередньо від організації збору, обробки, маршрутизації та розповсюдження інформації. Для побудови схеми руху інформації необхідно проаналізувати процеси її отримання, зберігання та використання під час роботи сервісного центру. На рисунку 1.2 зображено класифікацію інформаційних систем.

Комп’ютери, оснащені спеціалізованими програмними засобами, є технічною базою та інструментом для інформаційних систем. Інформаційна система немислима без персоналу, що взаємодіє з комп'ютерами та телекомунікаціями.

Інформаційні системи допомагають фахівцям, які працюють з даними, підвищують продуктивність та продуктивність роботи інженерів та проектувальників. Завдання подібних інформаційних систем – інтеграція нових відомостей в організацію та допомога в обробці паперових документів.

Рисунок 1.2 - Класифікація інформаційних систем

1.2 Сервісний центр обслуговування мобільних пристроїв

Спектр послуг на ринку ремонту мобільних пристроїв розширюється. З одного боку, це є свідченням розвитку та вдосконалення, з іншого боку – виникають проблеми з вибором, і допомогти у вирішенні цього питання може такий ресурс як сервісний центр. Послуги професійної допомоги, при купівлі, продажу та ремонті різних мобільних пристроїв та його комплектуючих об'єктивно користуються підвищеним попитом. Звідси висока конкуренція та

прагнення компаній різного формату завоювати довіру споживачів усілякими актуальними способами.

Створення сервісного центру – одна з найкращих можливостей успішно позиціонувати послуги в інтернеті, охопити максимум цільової аудиторії, підвищити якість сервісу, збільшити прибуток. Завдяки сайту сервісного центру фірма зможе заявити про себе, сформувати позитивну репутацію, розповісти мільйонам користувачів мережі про послуги та отримати ефективний інструмент інтернет-маркетингу.

Розробка інформаційної системи сервісного центру передбачає створення сучасного, якісного інтернет-майданчика зі зрозумілим інтерфейсом, привабливим дизайном, зручною системою управління, унікальним контентом, та побудова ефективного плану просування. Необхідно створити систему сервісного центру, яка повністю відображала б усі можливості та послуги, які надає дана фірма.

Значно покращують лояльність клієнтів додаткові сервіси, запроваджені під час замовлення інформаційної системи сервісного центру. Наприклад, клієнт зможе дізнатися, чи є в наявності потрібне обладнання або певна деталь та розрахувати вартість покупки, ремонту. Або ввівши особистий пароль отримати інформацію про те, на якому етапі продажу чи ремонту знаходиться його пристрій. Для підприємця-початківця достатня розробка ресурсу типу візитка, а для великих компаній з розширеною мережею центрів, розташованих у різних населених пунктах, оптимальним варіантом стане корпоративний сайт.

Віртуальне представництво в інтернеті значно мінімізує втрати тих клієнтів, які, побачивши традиційну рекламу, що не запам’ятали точну адресу майстерні або не додзвонилися за вказаними телефонами. Якісний вебзастосунок завжди працюватиме, надаючи повну інформацію та залучаючи клієнтів. Клієнтами компанії може бути як пересічні громадяни, і інші підприємства.

Важливим принципом роботи є розробка зручного інтернет-ресурсу з характерними елементами:

1) інтуїтивний інтерфейс;

2) якісний дизайн та контент;

3) простота управління.

Для роботи компанії потрібні професійні знання у влаштуванні приладів та сучасних технологій. Адекватне просування інтернет-ресурсу сервісного центру з ремонту мобільних пристроїв включає декілька завдань, які мають виконуватися одночасно. Трафік за запитами виявляється найскладнішим. Якщо сервісний центр позиціонує себе як сучасний та високотехнологічний, то відсутність у нього власного сайту може насторожити потенційних клієнтів, через що буде втрачено можливий прибуток. У свою чергу, професійно виконаний, зручний та приємний webзастосунок створює у відвідувачів позитивне враження та підвищить їх рівень довіри до сервісного центру. Під час роботи фахівці сервісного центру допомагають клієнтам у виборі товарів та послуг.

Про зв’язок клієнтів із сервісним центром варто поговорити докладніше. Сьогодні для забезпечення комунікацій цього необхідно наступне.

1. Телефон. Найкраще, якщо для сервісного центру буде використовуватись окремий номер або окремий додатковий номер. При цьому варто виділити спілкування з клієнтами центру окремої людини. Більшість запитів від покупців надходить саме по телефону, оскільки цей канал зв'язку — зручний звичним для багатьох і дає можливість отримати те саме живе спілкування з консультантом, яке так затребуване в будь-яких магазинах.

2. Email. Причому викладати адресу потрібно в зручній для покупця формі. Різні картинки та захищені від копіювання фрагменти тексту дуже дратують покупців. А тому краще створити спеціальну поштову скриньку для магазину і змиритися з деякою кількістю спаму, ніж втрачати потенційних клієнтів, які вважають за краще саме цей спосіб зв'язку.

3. Форма зворотного зв'язку на сторінці “Контакти”. Зручна альтернатива Email. При цьому на сторінці сайту розташована форма, в яку користувач може внести свої контактні дані (email, телефон), тему питання, саме повідомлення. І

після натискання кнопки “Надіслати” це повідомлення потрапить до пошти адміністратора сайту.

4. Онлайн-чат. Цей сервіс призначений для того, щоб покупець міг оперативно поставити запитання та отримати відповідь у текстовому вигляді. Сервіси онлайн-чатів зазвичай виглядають як спливаюче вікно "задати питання", вони дуже зручні і стають з кожним днем популярнішими.

Крім того, можливе розміщення додаткової інформації: фізичної та юридичної адреси, документів, сертифікатів, якщо це потрібно.

1.3 Аналіз існуючих інформаційних систем сервісних центрів

На сьогоднішній день існує досить велика кількість програм для автоматизації діяльності сервісних центрів. Найбільш популярними із них є РемОнлайн (https://remonline.app/, рис. 1.3), РемонтОнлайн, БазаКвитанций (https://ticketdb.ru/), WinService Pro, S-Center, MasterTool. Розглянемо їх функціонал докладніше.

РемонтОнлайн. Програма варта широкого використання, оскільки має великий набір різних типів обслуговування. Дозволяє обробляти замовлення, контролювати співробітників, вести складський облік, друкувати документи та звіти. Є елементи можливостей CRM, наприклад, автоматично відправляє SMS клієнту про завершення робіт над замовленням, а менеджера повідомляє по email про перебіг роботи над замовленням.

БазаКвитанций. Онлайн платформи для сервісних центрів. Прийом та робота з клієнтськими заявками. Складський облік та фінансова звітність. Друк необхідних документів. Інтеграція з сайтом та інтернет-магазином. Email інформування клієнтів.

WinService Pro (рис. 1.4). Програма обліку замовлень у сервісному центрі. Допомагає оформляти замовлення на приймання обладнання на ремонт від

клієнта, друкувати первинні документи, формувати необхідні звіти. Можливе автоматичне заповнення форм документів для подальшого друку.

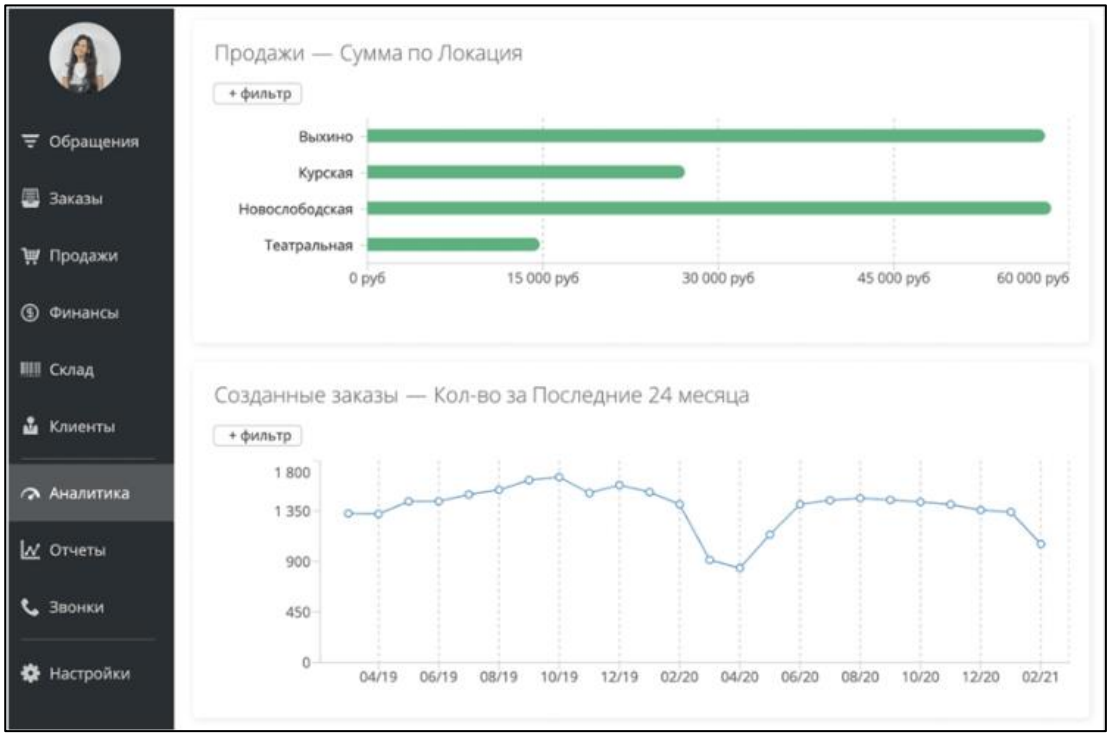

Рисунок 1.3 – Аналітика сервісного центру РемОнлайн

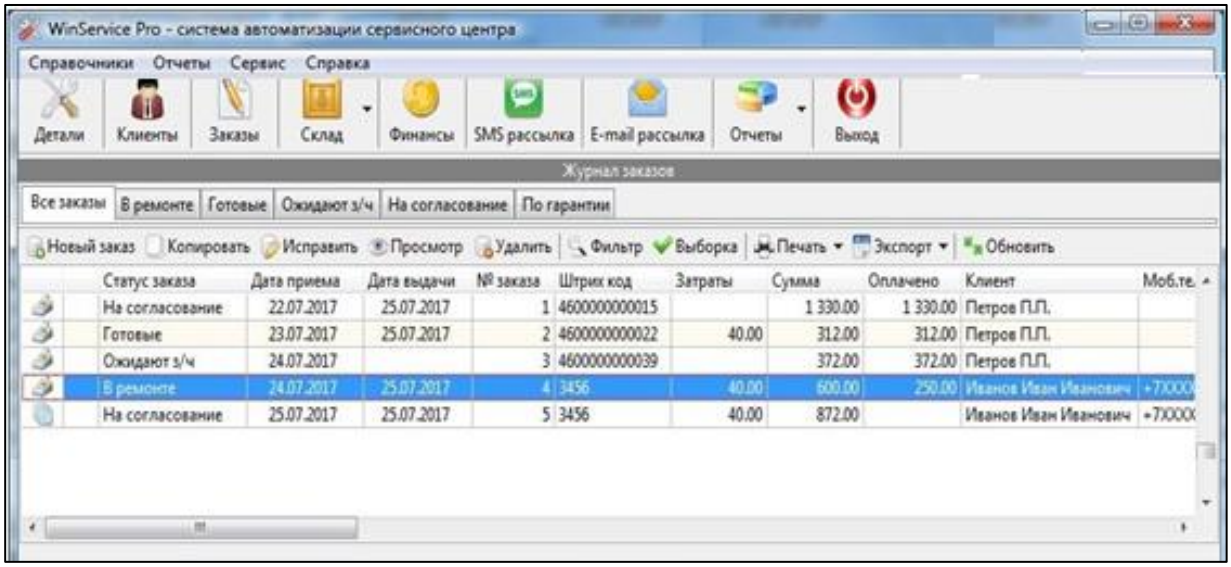

Рисунок 1.4 – Система автоматизованого сервісного центру WinService Pro

S-Center - програма для обліку повсякденної роботи сервісного центру призначена для обліку прийнятої в ремонт техніки, запасних частин на складі та відстеження виконаних робіт, а також видачі необхідних документів: акти приймання, акти виконаних робіт, гарантійні талони, копії чеків та звіти за статусами (рис. 1.5).

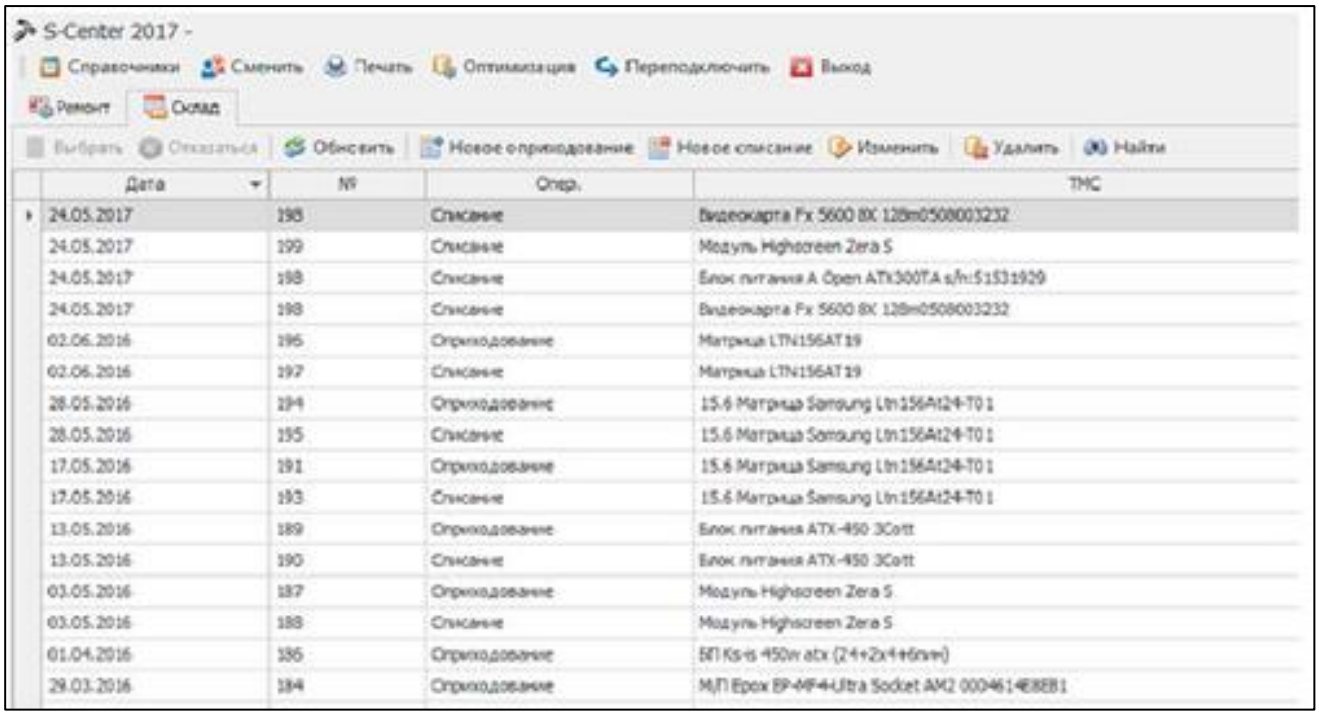

Рисунок 1.5 – Інтерфейс програми S-Center

MasterTool (рис. 1.6). Програма здатна враховувати об’єм виконаних робіт та затрачених на це матеріалів. Також відслідковуються прийняті на ремонт пристрої та обладнання в сервісному центрі чи майстерні.

Проаналізовані програмні продукти переважно мають схожі функції.

У кожній системі є можливості, затребувані будь-яким сервісним центром: облік замовлень, друк різних документів, облік деталей на складі, база клієнтів, формування різноманітних звітів.

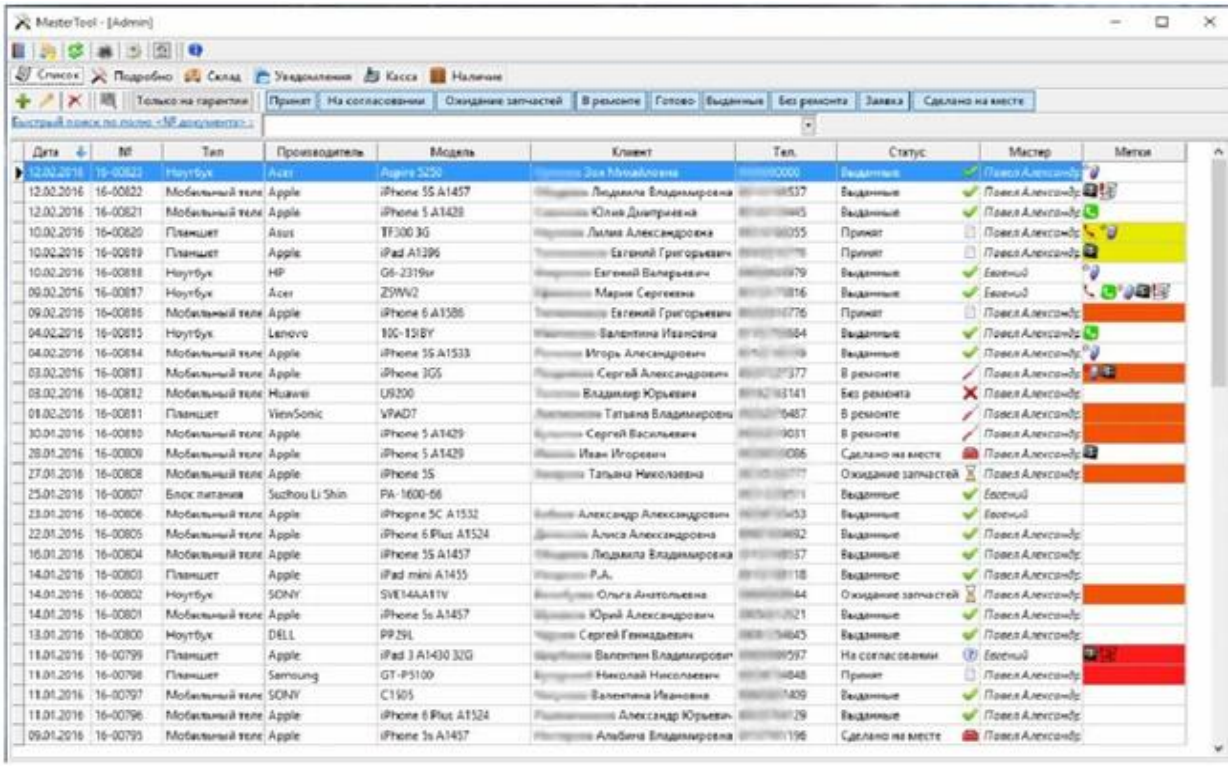

Рисунок 1.5 – Інтерфейс програми MasterTool

Здійснений аналіз можливостей популярних програмних рішень серед наявних інформаційних технологій, було виявлено наступне:

1) ціна багатьох програмних продуктів дуже часто висока для звичайного користувача;

2) обмежена кількість користувачів, які можуть одночасно використовувати систему, для збільшення необхідна доплата;

3) в окремих програмних продуктах обмежена кількість замовлень чи розмір бази даних;

4) практично немає можливості зміни функціоналу під себе, тому що немає відкритого коду;

5) багато функцій цих додатків стають доступними лише після додаткової плати.

Таким чином, розробка інформаційної системи сервісного центру прийому замовлень по обслуговуванню мобільних пристроїв дозволить поліпшити саме ці

аспекти технічного обслуговування мобільних пристроїв та покращити запити клієнтів на оформлення замовлень у режимі онлайн.

1.4 Постановка задачі дослідження

Розробка інформаційної системи прийому замовлень у сервіс-центрі по обслуговуванню мобільних пристроїв є актуальною проблемою на сучасному етапі інформатизації суспільства.

Об'єкт дослідження – сервісна технічна підтримка та обслуговування мобільних пристроїв.

Предмет дослідження – web-орієнтовані програмні засоби для обробки замовлень у сервіс-центрі по обслуговуванню мобільних пристроїв.

Метою даної роботи є поліпшення роботи сервісного центру по обслуговуванню мобільних пристроїв шляхом розробки і впровадження автоматизованої інформаційної системи прийому замовлень із можливістю відслідковування етапів ремонту у режимі реального часу.

Для досягнення поставленої мети було поставлено такі **завдання**:

– здійснити аналіз теоретичних засад діяльності центрів у сфері обслуговування мобільних пристроїв та виявити сучасний стан програмного забезпечення по прийому та обробці замовлень у сервісних центрах;

– дослідити та обґрунтувати вибір методу проєктування інформаційної системи;

– обґрунтувати вибір інструментальних засобів розробки та здійснити моделювання автоматизованої інформаційної системи прийому замовлень у сервіс-центрі;

– здійснити програмну реалізацію та тестування інформаційної системи прийому та обробки замовлень сервіс-центру з обслуговування мобільних пристроїв.

Висновки до розділу 1

Здійснений аналіз показав, що сучасний стан інформатизації суспільства супроводжується розширенням сфери інтернет-сервісу для підтримки технічного обслуговування мобільних пристроїв. Установлено, що сервісний центр є організацією, яка займається наданням послуг із сервісної підтримки та обслуговування техніки, обладнання та іншої продукції. Діяльність сервісних центрів включає перед торговий, гарантійний та після продажний ремонт і постачання виробленої продукції. Інформаційна система для підтримки діяльності сервіс-центру є сукупністю програмних та апаратних засобів, а також організаційного забезпечення, які надають інформаційну підтримку клієнтам та працівникам під час обслуговування мобільних пристроїв.

Здійснений аналіз показав, що в Україні існує багато інтернет-ресурсів для автоматизації діяльності сервісних центрів: РемОнлайн, РемонтОнлайн, БазаКвитанций, WinService Pro, S-Center, MasterTool. Вони мають широкий функціонал, який дозволяє контролювати співробітників, вести складський та фінансовий облік, формувати та друкувати документи і звіти, забезпечувати комунікацію співробітників центру з клієнтами. Проте багато з них є комерційними та дорогими й не надають можливості для зміни функціоналу під конкретну організацію. Для вирішення цієї проблеми є доцільним розробка інформаційної системи сервісного центру прийому замовлень по обслуговуванню мобільних пристроїв, яка дозволить клієнтам оформляти замовлень у режимі онлайн.

2 МЕТОДИ ПРОЄКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

Одним із базових понять проєктування інформаційних систем є поняття життєвого циклу проєкту, який визначається моделлю й описується у формі методу – методології, яка визначає комплекс робіт, їх послідовність, детальний зміст та відповідальність фахівців на усіх етапах розробки ІС.

2.1 SADT-методологія

Найпоширенішою технікою проектування інформаційних систем є методологія структурного аналізу SADT (Structured Analysis and Design Technique) [3]. Вона є набором методів, нотацій і правил, що описують процедури побудови функціональної моделі об’єкта або процесу. Функціональна модель, побудована в SADT показує структуру функцій об’єкта. Дії, що виробляються об'єктом та взаємозв'язки між цими діями. Кожна функція нижчого порядку може використовувати лише ті елементи, які входили до функцій вищого порядку, крім першої функції найвищого порядку, куди будуть поставлятися вихідні дані. SADT-діаграми можуть не вказувати спеціально конкретний порядок виконання та час. На діаграмі можна використовувати кілька видів зв’язків:

1) проста;
2) логічна;
3) процедурна;
4) функціональна;
5) послідовна.

Зазвичай SADT-методологія застосовується на ранніх етапах життєвого циклу інформаційної системи. - модель - це точний, повний і адекватний текстовий і графічний опис системи, що має конкретне призначення, виконане у вигляді ієрархічно організованої сукупності діаграм, створених на основі

стандартного представлення даних. Це опис системи, яка має єдиний суб'єкт та мету.

Така модель являє собою сукупність ієрархічно впорядкованих та взаємопов'язаних діаграм, організованих у вигляді деревоподібної структури, де верхня діаграма є найбільш загальною, а найнижчі найбільш деталізовані.

У SADT-моделях використовуються як натуральна, так і графічна мови. Для передачі інформації про конкретну систему джерелом натуральної мови служать люди, що описують систему, а джерелом графічної мови - сама методологія SADT. Графічна мова SADT забезпечує структуру та точну передачу моделі семантики природної мови. З точки зору SADT, модель може бути зосереджена або на функціях системи, або на її об'єктах.

2.2 IDEF0-методологія

На основі SADT у 1981 році була створена методологія моделювання IDEF0 (Icam DEFinition). Вона призначена для аналізу всієї системи як множини взаємопов'язаних, взаємодіючих функцій. Орієнтація виключно на аналіз функцій дозволяє розглядати функції незалежно від об'єктів, що їх виконують [4].

Модель, побудована з цієї нотації представляє набір ієрархічно упорядкованих і навіть об'єднаних у порядку діаграм. Кожна діаграма вважається цілісною сутністю в системі на одній спільній діаграмі, при цьому кожна з них окремо може бути відредагована відносно незалежно.

Функціональний підхід дозволяє чітко визначити проблеми аналізу та проектування від проблем реалізації. Опис системи за правилами IDEF0 має чітку структуру. IDEF0 — модель представляє собою ієрархічно впорядковані діаграми (рис. 2.1). Кожна діаграма описує певну функцію і складається з кількох взаємодіючих та взаємопов'язаних підфункцій, кожна з яких у свою чергу може бути описана діаграмою.

Розробка моделей IDEF0 вимагає дотримання ряду строгих формальних правил, які забезпечують переваги методології щодо однозначності, точності та цілісності складних багаторівневих моделей.

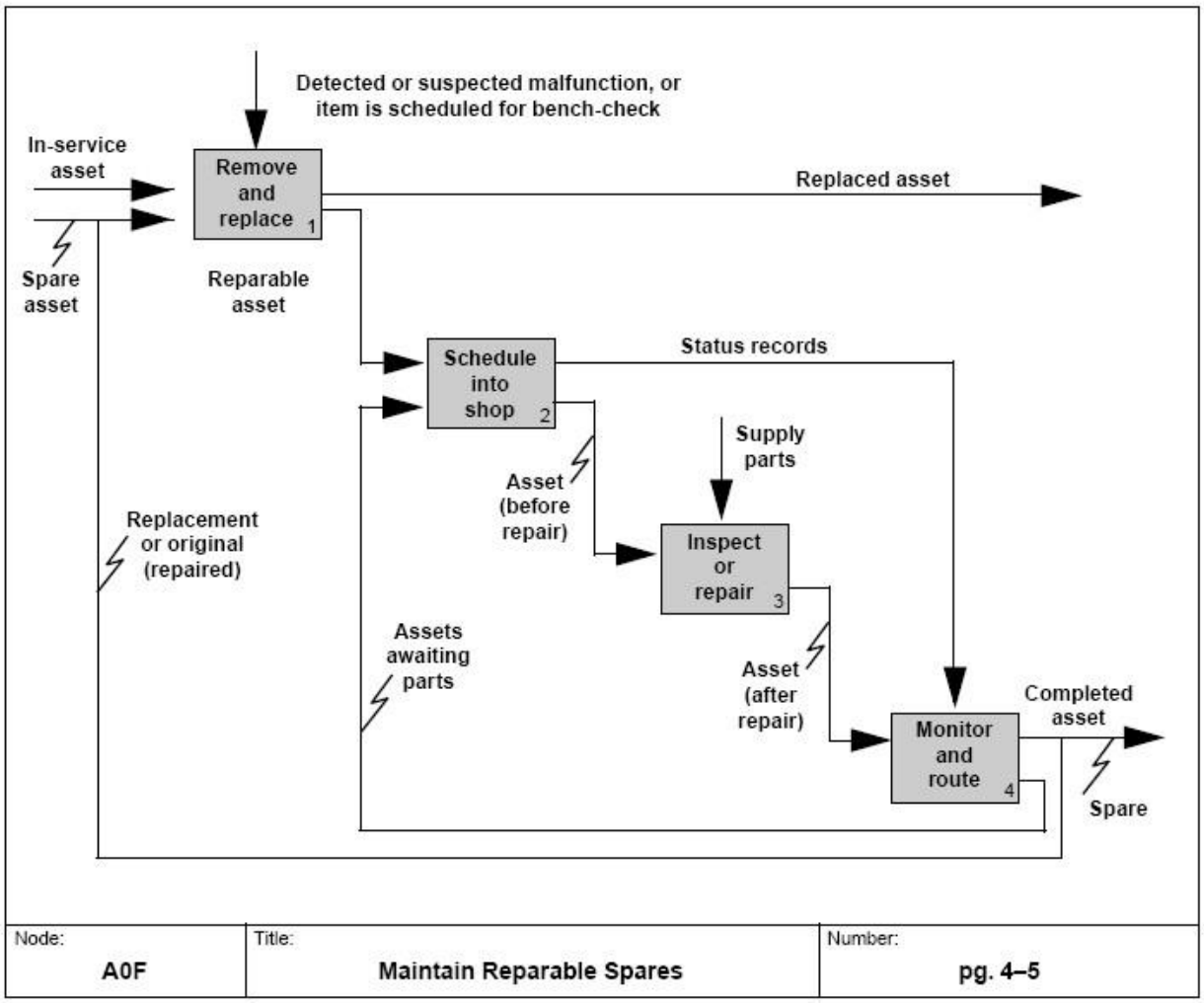

Рисунок 2.1 – Приклад IDEF0 діаграми

2.3 DFD-методологія

При побудові функціональної моделі системи альтернативою методології SADT (IDEF0) є методологія діаграм потоків даних (Data Flow Diagrams, DFD). На відміну від IDEF0, що призначена для проектування систем взагалі, DFD призначена для проектування інформаційних систем.

Елементів побудови не так багато, як у IDEF0, що так само для когось може бути одним із плюсів через простоту роботи малою кількістю основних елементів. Усього є чотири елементи (рис. 2.2):

1) зовнішня сутність приймає та надсилає дані із загальної системи;
2) процес – це обробник, який змінює дані будь-яким чином, або перенаправляє їх;
3) сховище даних містить інформацію, але з обробляє її;
4) потік даних показує переміщення даних між іншими елементами.

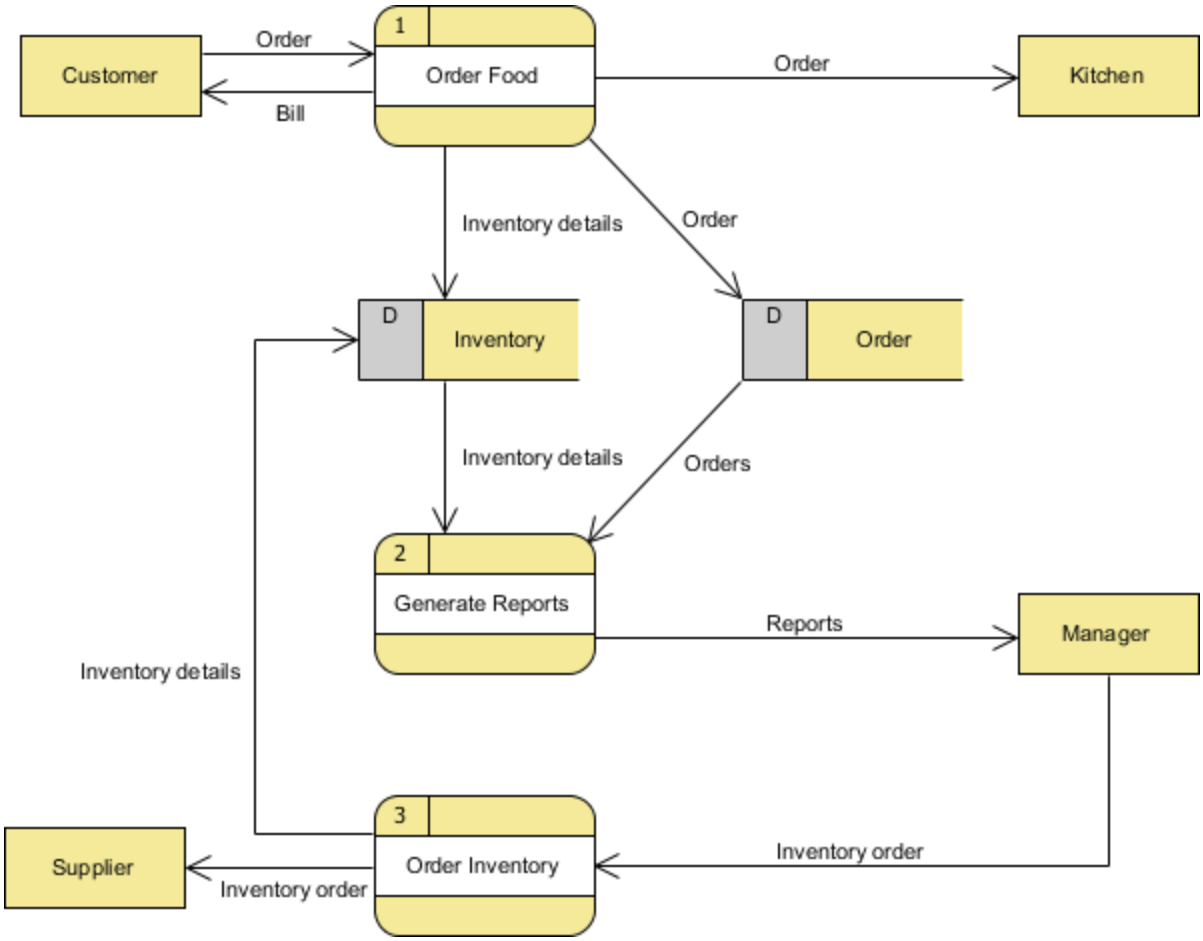

Рисунок 2.2 – Приклад DFD діаграми

Діаграми, побудовані за такою методологією, відображають карту потоків інформації для бізнес-процесу або інформаційної системи. При цьому всі вони дотримуються певного набору правил [5]:

1) кожен процес повинен мати хоча б один вхідний потік і хоча б один вихідний;
2) кожне сховище даних повинно мати хоча б один вхідний потік і хоча б один вихідний;
3) дані, що зберігаються в системі, повинні були пройти як мінімум один процес обробки.

2.4 Створення інформаційної системи на основі каскадної моделі

Стандарт ГОСТ 34.601–90 описує стадії та етапи проектування ІС на основі використання каскадної моделі. Стадії та етапи робіт можуть мати різну трудомісткість.

Це залежить від складності об'єкта автоматизації і набору завдань для вирішення при створенні конкретної ІС. На будь-якій стадії проекту допускається:

1) об'єднувати послідовні етапи і вилучати деякі з них;

2) розпочинати виконання робіт наступної стадії до закінчення попередньої.

Стадії та етапи створення ІС, які виконуються організаціями-учасниками, прописуються в договорах і технічних завданнях на виконання робіт:

Перша стадія. Формування вимог до ІС. На першій стадії проектування виділяють наступні етапи:

1) оцінка об'єкта інформатизації та визначення необхідності створення ІС;

2) створення списку вимог користувачів до Інформаційної Системи;

3) оформлення звіту про виконану роботу і технічне завдання на розробку.

Друга стадія. Розробка концепції ІС. Ця стадія поділяється на такі етапи робіт:

1) визначення об'єкта автоматизації;

2) виконання науково-дослідних робіт, що необхідні;

3) розробка концептуальних прикладів ІС, які задовольняють вимогам користувачів;

4) оформлення звіту та затвердження концепції.

Третя стадія. Технічне завдання. Ця стадія включає розробку та затвердження ТЗ на створення ІС.

Четверта стадія. Ескізний проект. Ця стадія поділяється на такі етапи робіт:

1) розробка проектних рішень по системі та по її частинам;

2) розробка ескізної документації на ІС та на її частини.

П'ята стадія. Технічний проект. Ця стадія поділяється на наступні етапи робіт:

1) розробка документації на поставку комплектуючих;

2) постановка завдань на проектування в суміжних частинах ІС.

Шоста стадія. Робоча документація. На цій стадії виділяють такі етапи робіт:

1) розробка робочої документації на ІС та її частини;

2) розробка та адаптація програм.

Сьома стадія. Введення в дію. Ця стадія включає наступні етапи робіт:

1) підготовка об'єкта автоматизації;

2) підготовка персоналу;

3) комплектація ІС виконується програмними і технічними засобами, програмно–технічними комплексами, інформаційними виробами;

4) будівельно-монтажні роботи;

5) пуско-налагоджувальні роботи;

6) проведення попередніх випробувань та дослідної експлуатації;

7) проведення приймальних випробувань.

Восьма стадія. Супровід ІС. Ця стадія включає такі етапи робіт:

1) виконання робіт відповідно до гарантійних зобов'язань;

2) післягарантійне обслуговування.

Висновки до розділу 2

У другому розділі було установлено, що одним із базових понять проєктування інформаційних систем є поняття життєвого циклу проєкту, який визначається моделлю й описується у формі методу – методології, яка визначає комплекс робіт, їх послідовність, детальний зміст та відповідальність фахівців на усіх етапах розробки ІС.

Здійснено аналіз існуючих підходів до проектування інформаційної системи та виявлено найбільш поширені методології: SADT: методологія структурного аналізу, IDEF0: функціонального моделювання, DFD: діаграм потоків даних та каскадна модель. У результаті здійсненого аналізу обґрунтовано вибір методу побудови ІС на основі каскадної моделі та виявлено основні етапи цього методу: 1) формування вимог до ІС, 2) розробка концепції ІС, 3) розробка та затвердження ТЗ на створення ІС, 4) розробка ескізного проекту, 5) розробка технічного проекту, 6) розробка робочої документації, 7) впровадження розробленої ІС, 8) супровід ІС: гарантійне та післягарантійне обслуговування.

3 ІНСТРУМЕНТАЛЬНІ ЗАСОБИ РОЗРОБКИ ТА МОДЕЛЮВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

3.1 Архітектура застосунку

Архітектура програмного забезпечення зазвичай відноситься до більших структур програмної системи, і вона стосується того, як кілька процесів програмного забезпечення взаємодіють для виконання своїх завдань. Архітектура системи описує її основні компоненти, їх відносини (структури) та те, як вони взаємодіють один з одним. Архітектура програмного забезпечення включає ряд факторів, таких як бізнес-стратегія, атрибути якості, динаміка людини, дизайн і IT-середовище.

Архітектура є основою системи. Вона забезпечує абстрагування для управління складністю системи та створення механізму зв'язку та координації серед компонентів. Вона визначає структуроване рішення для задоволення всіх технічних та експлуатаційних вимог, одночасно оптимізуючи загальні атрибути якості, такі як продуктивність та безпека [16].

Крім того, вона включає ряд важливих рішень щодо організації, пов'язаної з розробкою програмного забезпечення, і кожне з цих рішень може вплинути на якість, розширюваність, продуктивність і загальний успіх кінцевого продукту.

Найбільш поширеним шаблоном архітектури є багаторівневий, інакше відомий як шаблон архітектури n-рівня. Ця модель є стандартом для більшості програм Java EE і тому широко відома більшості архітекторів, дизайнерів та розробників. Структура багаторівневої архітектури тісно пов'язана з традиційними IT-комунікаціями та організаційними структурами, виявленими у більшості компаній, що робить її природним вибором для більшості зусиль з розробки бізнес-застосунків. Компоненти в структурі багаторівневої архітектури організовані в горизонтальні шари, причому кожен шар виконує певну роль усередині програми (наприклад, логіка уявлення чи бізнес-логіка). Незважаючи

на те, що в шаблоні багаторівневої архітектури не вказується число та типи шарів, які повинні існувати в шаблоні, більшість шаруватих архітектур складаються із чотирьох стандартних шарів: презентаційний шар, бізнес, шар взаємодії з базою даних та бази даних. У деяких випадках бізнес-рівень та рівень взаємодії з базою даних поєднуються в єдиний бізнес-рівень, зокрема, коли логіка взаємодії з базою даних (наприклад, SQL або HSQL) знаходиться у бізнес-рівнях. Таким чином, маленькі програми можуть мати тільки три шари, тоді як більші і складніші бізнес-програми можуть містити п'ять або більше шарів.

Кожен рівень шаблону багаторівневої архітектури відіграє певну роль та відповідальність у рамках програми. Наприклад, презентаційний шар відповідатиме за обробку всього інтерфейсу користувача та логіки зв'язку з браузером, тоді як бізнес-рівень буде відповідальним за виконання конкретних бізнес-правил, пов'язаних із запитами. Кожен шар в архітектурі утворює абстракцію навколо роботи, яка має бути виконана для задоволення певного процесу бізнесу.

Однією з найпотужніших особливостей багаторівневої архітектури є поділ проблем між компонентами. Компоненти всередині певного рівня відноситься тільки до логіки, яка відноситься до цього шару.

Для розроблюваної інформаційної системи було обрано трирівневу архітектуру застосунку, зображену на рисунку 3.1.

Перший рівень це клієнтська частина, до неї входить інтерфейс користувача і вся необхідна логіка для організації взаємодії користувача з системою.

Другий рівень це сервер, на ньому буде реалізована основна логіка програми: обробка даних, генерація звітів, взаємодія з різними системами.

Третій рівень це база даних, у якій зберігаються всі необхідні для роботи програми.

Після того, як визначена архітектура програми, необхідно вибрати мовні та інструментальні засоби розробки.

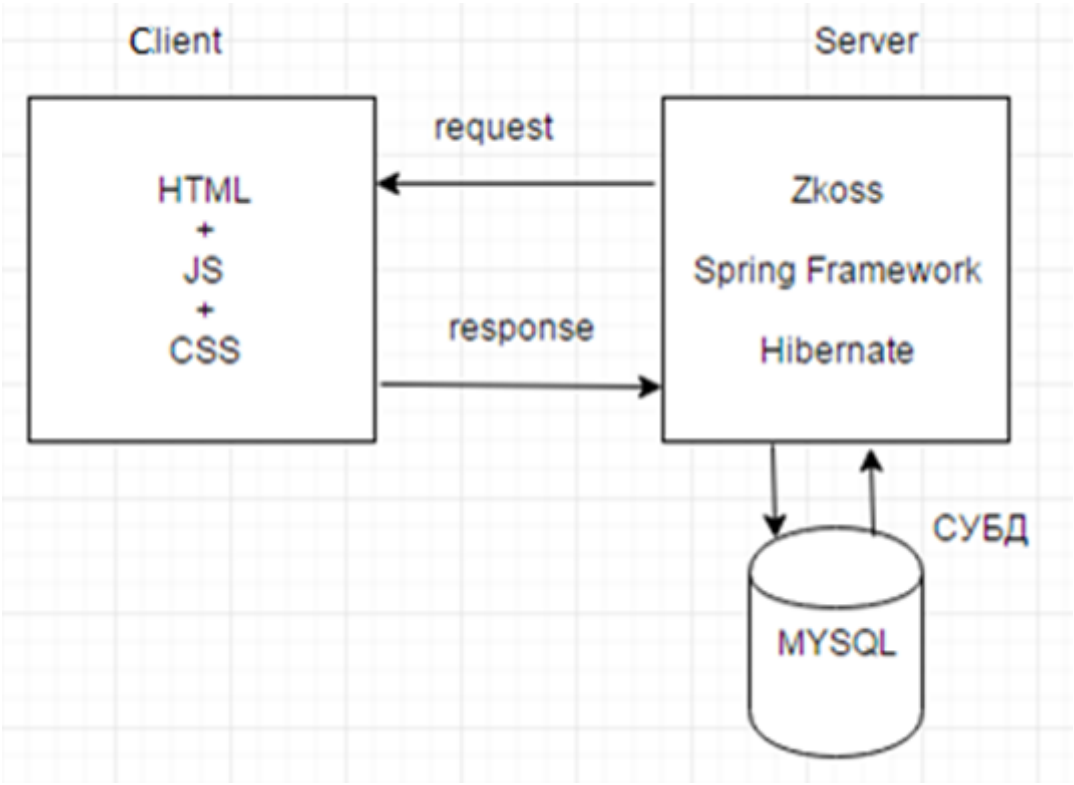

Рисунок 3.1 – Архітектура застосунку

3.1 Технології та інструментальні засоби розробки

Для вирішення поставленого завдання необхідно визначитися з інструментами та технологіями, за допомогою яких проводитиметься розробка даної системи. Оскільки однією з головних вимог є доступ до системи через інтернет браузер, ми змушені розробляти web-застосунок. Використання CMS для розробки даної програми неможливо, оскільки всі CMS мають обмежений функціонал і програми, що мають складну бізнес логіку, важко розробляються за допомогою таких систем. Тому подібні програми здебільшого розробляються самостійно за допомогою високорівневих мов програмування.

Оскільки користувач для роботи з додатком використовуватиме інтернет браузер, то логічно використовувати на клієнтській частині зв'язку HTML, CSS і JavaScript. HTML – стандартизована мова розмітки документів, вона дозволяє легко розробляти інтерфейс користувача. CSS - формальна мова опису

зовнішнього вигляду документа, написаного за допомогою мови розмітки. JavaScript – це об'єктно орієнтована мова програмування, що часто використовується для формування логіки клієнтської частини програми. Ця зв'язка є стандартом при розробці веб-додатків.

Для розробки серверної частини даної системи було обрано мову Java - типізована об'єктно-орієнтована мова програмування, розроблена компанією Sun Microsystems. Ця мова була обрана з низки причин [7]:

– кросплатформність забезпечує переносимість додатків на різні платформи;
– строга типізація;
– наявність прибирача сміття дозволяє при розробці програми не витрачати час на виявлення та виправлення витоків пам'яті;
– велика кількість інформації щодо тонкощів мови;
– сильно розвинене інтернет співтовариство.

Ці та інші переваги даної мови дозволяють досить швидко розробляти різні програми.

При виборі СУБД необхідно дотримуватися кількох критеріїв: безкоштовне розповсюдження, підтримка виробника, зручність та простота у роботі, наявність інформації в інтернеті, надійність, продуктивність.

Під цей критерій потрапили такі системи управління базами даних: Firebird (FirebirdSQL), H2, MySQL, PostgreSQL. Для розробки програми вирішено було використовувати MySQL, так як це найбільш знайома СУБД для розробника. Розробку та підтримку MySQL здійснює корпорація Oracle. MySQL є рішенням для малих та середніх додатків. До того ж ця СУБД дуже популярна. Має зручний інструмент для розробки mysql workbench. Він дозволяє без особливих зусиль створити схему бази даних, написати та виконати будь-які запити до бази даних [8].

Для того, щоб найбільш швидко розробляти клієнтську частину програми, було прийнято рішення використовувати генератори HTML сторінок. До переваг такого підходу можна віднести [9]:

— швидка розробка інтерфейсу користувача;

— наявність готових візуальних компонентів;

— гнучка можливість інтегрувати стандартні, сторонні або власні програмні компоненти в сторінки.

Основним недоліком цього підходу є збільшення навантаження на сервер. У мережі інтернет знайдено такі технології: Apache Wicket, Tapestry, OpenLaszlo, ZK. Всі представлені Фреймворки дозволяють створювати інтерфейси для веб додатків, використовуючи для вирішення більшості завдань мову Java.

Apache Wicket - фреймворк із відкритим вихідним кодом для створення веб-застосунків. Розроблено Джонатаном Локе у 2004 році. З червня 2007 є проектом Apache Software Foundation. Для використання даної технології потрібне знання HTML та JAVA, можливе використання технології AJAX без написання коду мовою JavaScript [10].

OpenLaszlo – платформа з відкритим вихідним кодом, призначена для розробки та доставки багатофункціональних інтернет-додатків. OpenLaszlo – це документ.lzx з XML подібним синтаксисом.

Tapestry – фреймворк для створення веб-застосунків, що реалізують шаблон MVC («Модель-представлення-контролер»). Tapestry був створений Говардом Льюїсом Шипом і продовжує активно розвиватися. Фреймворк дозволяє легко використовувати технологію AJAX, валідацію даних, надає можливість локалізації веб-застосунків.

ZK - фреймворк для розробки web-застосунків, є як безкоштовна, так і платна версії, що дозволяє полегшити розробку інтерфейсу програми. Має XML подібний синтаксис [11].

ZK надійний і створений із солідним послужним списком як основа вибору у різних галузях. Десятки тисяч розробників використовують ZK для створення

своїх критично важливих систем, включаючи багатомільйонні системи, що обслуговують мільйони користувачів та десятки тисяч паралельних сесій у міжнародному масштабі.

Файли, що використовуються для представлення користувачеві мають розширення *.zul.

У наведеній нижче таблиці відображені результати порівняння даних фреймворків.

Таблиця 3.1 – Порівняння генераторів HTML сторінок

Критерій	Apache Wicket	OpenLaszlo	Tapestry	ZK
Наявність безлічі готових компонентів	-	-	-	+
Можливість керувати вмістом сторінки за допомогою Java коду	+	+	+	+
Відсутність необхідності знати HTML	-	+	+	+
Взаємодія за шаблоном MVVM	-	-	-	+
Наявність великої кількості прикладів використання	+	+	-	+

Серед представлених фреймворків для вирішення завдання взаємодії з користувачем було обрано технологію ZK. Тому представимо її більш докладний опис.

Zkoss перетворює файли формату *.zul на html документ за допомогою внутрішньої логіки фреймворку. Синтаксис мови відрізняється від html, проте підтримує усі його конструкції. Підтримує мову JavaScript та ZScript. Є підтримка CSS3 та ZCSS. Zkoss сам здійснює формування запиту на сервер, а також обробку даних для відображення користувачеві.

Дозволяє працювати за двома шаблонами проектування MVC та MVVM. У розроблюваному застосунку використовується другий спосіб, тому що є більш продуктивним і більш простим у реалізації.

На рисунку 3.2 представлений принцип роботи програми шаблону MVVM.

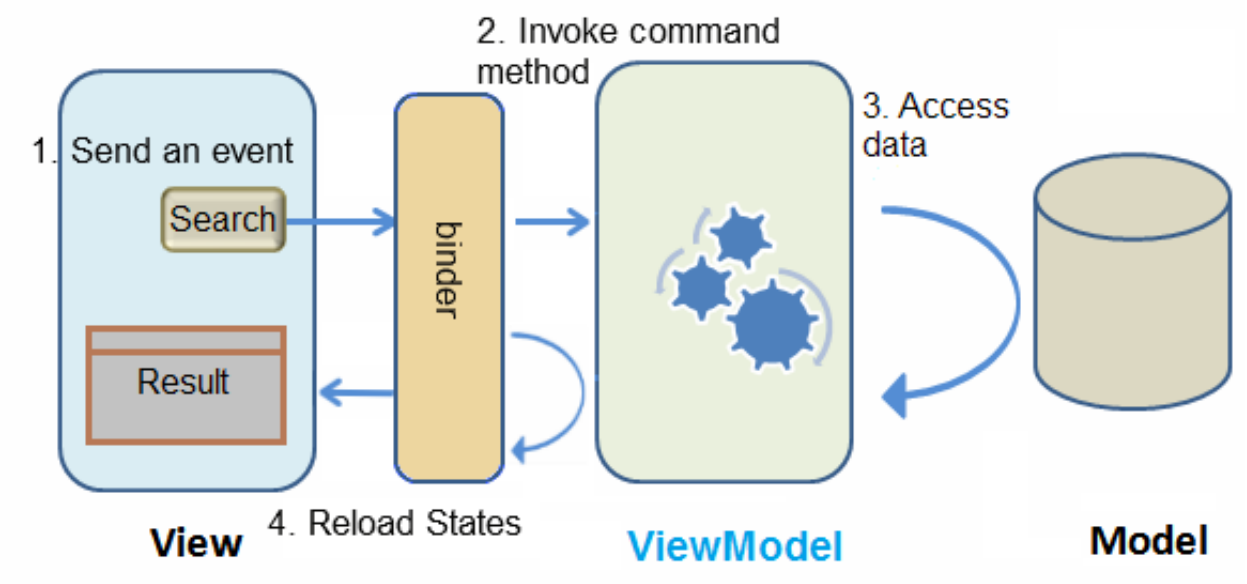

Рисунок 3.2 - Принцип роботи програми за шаблоном MVVM

Також Zkoss підтримує інший, мабуть, найпопулярніший фреймворк у мові java Spring. У всіх ViewModel використовуваних Zkoss необхідно використовувати замість інструкції @Autowired або @Inject інструкцію @WireVariable або @Wire.

Spring Framework - універсальний фреймворк з відкритим вихідним кодом для Java-платформи. Spring забезпечує вирішення багатьох завдань, з якими стикаються Java-розробники та організації, які хочуть створити інформаційну систему, засновану на платформі Java. Spring не повністю пов'язаний із платформою Java Enterprise, незважаючи на його масштабну інтеграцію з нею, що є важливою причиною його популярності [12].

Для поділу повноважень користувачів та забезпечення безпеки потрібна авторизація. Кожен користувач має унікальний логін та пароль. Кожному користувачеві призначаються певні права доступу до даних. Дані про користувача та ролі зберігаються в базі даних. Як інструмент для забезпечення безпеки вибраний Spring Security. Цей фреймворк надає комплексні служби безпеки для програм корпоративного ПЗ на базі Java EE. Особливу увагу приділяють підтримці проектів, створених з використанням Spring Framework, який є провідним рішенням Java EE для розробки корпоративного програмного забезпечення.

Для організації взаємодії бази даних та програми у мові Java існує спеціальна специфікація - JPA (Java Persistence API). Ця специфікація визначає систему управління збереження Java об'єктів у таблиці реляційних баз даних у зручному вигляді. У мові немає передбаченої реалізації даної специфікації, проте існує безліч реалізацій даної специфікації від різних компаній. Даний спосіб збереження Java об'єктів у базі даних є найпопулярнішим, саме він і використовується у цьому проекті.

Як реалізацію специфікації JPA використовується Hibernate - бібліотека для мови програмування Java, призначена для вирішення завдань об'єктно-реляційного відображення (ORM). Є найпопулярнішим фреймворком для роботи з базами даних [13].

Для автоматизованого виконання рутинних операцій таких як компіляція, збирання, виконання тестів та додавання бібліотек до проекту необхідно використовувати спеціалізований фреймворк збирач. До найбільш популярних

збирачів відносяться: Apache Maven, Gradle, Apache Ant. Порівняння даних фреймворків представлені у таблиці 3.2.

Оскільки з усіх представлених засобів для складання проекту найбільшою популярністю користується Apache Maven, то він і буде використовуватися в цьому проекті.

Таблиця 3.2 – Порівняння засобів автоматичної зборки проекту

Критерій	Apache Maven	Gradle	Apache Ant
Незалежність від OS	+	+	+
Управління залежностями	+	+	-
Можливість збирання з командного рядка	+	+	+
Інтеграція із середовищами розробки	+	+	-

Web-програми, що розробляються на Java, працюють за допомогою Java Servlet API. Сервлет є інтерфейсом Java, реалізація якого розширює функціональні можливості сервера. Сервлет взаємодіє з клієнтами за допомогою принципу запит-відповідь. Загальна взаємодія клієнта та сервлета виглядає так: після того, як користувач надіслав запит веб-серверу, сервер надсилає запит контейнеру сервлетів. Контейнер сервлетів виконує запит і надсилає відповідь веб-серверу як html сторінку [14].

Для роботи програми потрібен контейнер сервлетів та веб-сервер. Apache Tomcat може одночасно використовуватися як контейнер сервлетів та веб-сервер. Використання Apache Tomcat є стандартом при розробці веб-застосунків з використанням Java Servlet API. Порівняно з Jetty працює швидше.

Для того щоб у процесі розробки не було втрачено вихідний код проекту, необхідно використовувати систему контролю версій. Система контролю версій дозволяє вести історію змін, вести розробку командою, оперативно взаємодіяти з віддаленими репозиторіями.

Серед популярних систем контролю версій можна назвати такі: GIT, SVN (Subversion), Mercurial. Усі представлені системи мають свої переваги та недоліки. У цьому проекті використовується GIT оскільки він, у порівнянні з іншими системами контролю версій працює із змінами, а не з файлами та є розподіленої системою [15]. Працювати з GIT набагато простіше, ніж з SVN та Mercurial, до того ж GIT відмінно взаємодіє з редакторами, такими як Intellij Idea та Eclipse.

3.3 Проєктування структур даних, алгоритмів та дизайну інтерфейсу

Алгоритм є покроковою процедурою, яка визначає набір команд, які повинні виконуватися в певному порядку, щоб отримати бажаний результат.

Структура даних є компонуванням даних у пам'яті комп'ютера або навіть на диску. Прикладом кількох загальних структур даних є масиви, пов'язані списки, черги, стеки, бінарні дерева та хеш-таблиці. З іншого боку, алгоритми використовуються для управління даними, що містяться в цих структурах даних, як при пошуку та сортуванні. Багато алгоритмів застосовуються безпосередньо до конкретних структур даних. При роботі з певними структурами даних необхідно знати, як вставляти нові дані, шукати певний елемент та видаляти певний елемент.

Для початку розробки вебзастосунку необхідно встановити таке програмне забезпечення: GIT, Apache Maven, Intellij Idea, Tomcat 7.0. JDK 1.8. При запуску Intellij Idea вказуємо розташування JDK та Apache Maven та створюємо порожній maven проект. У файлі pom.xml вказуємо необхідні залежності: ZK, Spring, Spring Security, JPA, Hibernate.

Apache Maven перед запуском програми автоматично завантажує з віддаленого репозиторію необхідні бібліотеки. Нам лише необхідно вказати версію необхідного нам фреймворку, groupId – найменування організації або підрозділу, artifactId – назва проекту. Також якщо є необхідність вказується Оскільки один фреймворк може мати велику кількість бібліотек необхідних для розробки, а версії бібліотек повинні збігатися, для уникнення різноманітних

конфліктів, зручнішого управління версіями бібліотек і уникнення дублювання інформації версії бібліотек прописуються в змінні.

У Intellij Idea є панель управління збирачем maven, в якій представлені основні команди для роботи з даним інструментом. Після натискання кнопки “Reimport All Maven projects” відбудеться автоматичне завантаження необхідних бібліотек та видалення непотрібних.

Дизайн інтерфейсу користувача (UI) або розробка інтерфейсу – це розробка інтерфейсів для машин і програмного забезпечення, таких як комп'ютери, побутова техніка, мобільні пристрої та інші електронні пристрої, з акцентом на максимізацію зручності використання.

Мета дизайну інтерфейсу користувача – зробити взаємодію користувача максимально простою і ефективною з точки зору досягнення цілей користувача (орієнтований на користувача дизайн). Хороша конструкція інтерфейсу користувача полегшує виконання завдання під рукою, не звертаючи на себе зайвої уваги. Графічний дизайн та друкарня використовуються для підтримки його зручності використання, впливаючи на те, як користувач виконує певні взаємодії та покращує естетичну привабливість дизайну.

Дизайн може посилити або зменшити здатність користувачів використовувати функцію інтерфейсу. Процес проектування повинен збалансувати технічну функціональність і візуальні елементи (наприклад, ментальну модель), щоб створити систему, яка не тільки працездатна, але й придатна для використання та адаптується до потреб користувачів. Дизайн інтерфейсу пов'язаний із широким спектром проектів від комп'ютерних систем до автомобілів, комерційних літаків; Всі ці проекти пов'язані з однією і тією самою основною людською взаємодією, але також потребують деяких унікальних навичок та знань. В результаті розробники схильні спеціалізуватися на певних типах проектів і мати навички, орієнтовані на їх досвід, будь то розробка програмного забезпечення, дослідження користувачів, веб-дизайн або промисловий дизайн.

Висновки до розділу 3

У третьому розділі було з’ясовано, що використання CMS для розробки інформаційної системи не є доцільним, оскільки всі CMS мають обмежений функціонал і не дозволяють повноцінно реалізувати складну бізнес логіку. Було вирішено розробляти web-застосунок із трирівневою архітектурою: 1) клієнтська частина: включає інтерфейс користувача і організацію взаємодії користувача з системою; 2) серверна частина: включає сервер, де реалізовано основну логіка програми: обробку даних, генерацію звітів, взаємодію з різними системами; 3) база даних: містить усю інформацію стосовно роботи системи. Було побудовано архітектуру застосунку та розглянуто шаблони проектування і побудови HTML-сторінок.

Для розробки клієнтської частини web-застосунку було обрано HTML, CSS, JavaScript та фреймворк ZK, який дозволяє працювати з обраним шаблоном проєктування MVVM. Для розробки серверної частини обрано мову програмування Java, інтегроване середовище розробки IntelliJ Idea, пакет JDK та фреймворк з відкритим кодом Spring Framework. Для забезпечення безпеки обрано фреймворк Spring Security.

Для отримання й зберігання даних обрано базу даних MySQL, яка має зручний інструмент для розробки MySQL Workbench, що полегшує створення схеми бази даних та виконання запитів до неї. Організацію взаємодії бази даних та програми на мові Java реалізовано з використанням бібліотеки Hibernate. Для автоматизованого виконання операцій компіляції, збирання, виконання тестів та додавання бібліотек до проєкту використано спеціалізований фреймворк збирач Apache Maven. При розробці було також вирішено також використовувати контейнер сервлетів та веб-сервер Apache Tomcat і систему контролю версій GIT.

У розділі також описано послідовність застосування вказаних інструментальних засобів при розробці вебзастосунку.

.

4 ПРОГРАМНА РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ ПРИЙОМУ ЗАМОВЛЕНЬ У СЕРВІС-ЦЕНТРІ ПО ОБСЛУГОВУВАННЮ МОБІЛЬНИХ ПРИСТРОЇВ

4.1 Особливості реалізації

Цей етап також відомий як фаза програмування. Реалізація дизайну програмного забезпечення починається з точки зору написання програмного коду відповідною мовою програмування та ефективного розгортання виконуваних програм без помилок. Мета етапу реалізації полягає в тому, щоб перевести дизайн системи на код заданою мовою програмування. Для цього проекту метою цього етапу є найкраща реалізація конструкції.

Фаза написання коду впливає як на тестування, так і на підтримку. Добре написаний код знижує витрати на тестування та технічне обслуговування. Оскільки вартість тестування та обслуговування програмного забезпечення є набагато вищою, ніж вартість написання коду, метою має бути скорочення зусиль з тестування та обслуговування. Отже, під час написання коду основну увагу слід приділяти розробці програм, які легко писати. На цьому етапі слід прагнути до простоти та ясності. Оскільки цей проект має багаторівневу архітектуру, нижче докладніше розказано про створення кожного рівня програмного продукту.

4.2 Розробка структури бази даних

Як було сказано раніше, у якості СУБД було прийнято рішення використовувати MySQL. У базі даних ми зберігатимемо всю інформацію, необхідну для забезпечення роботи сервісного центру.

Таблиця USERS зберігає інформацію про користувачів системи, містить такі поля:

– ID INT(11) – ідентифікаційний номер;

– NAME VARCHAR(45) – ім'я користувача;
– SECOND_NAME VARCHAR(45) – по батькові;
– FAMILY_NAME VARCHAR(45) - прізвище;
– EMAIL VARCHAR(45) – електронна пошта;
– PASSWORD – пароль для доступу до системи зберігається у зашифрованому вигляді;
– PLACE_ID INT(11) – ІД місця якого прив'язаний користувач системи.

Кожен користувач має певний набір прав у системі, повний список прав зберігатиметься в таблиці RIGHTS:

– ID INT(11) – ідентифікаційний номер;
– TYPE VARCHAR(45) – тип ресурсу у системі;

таблиці USERS та RIGHTS мають зв'язок багато до багатьох.

Кожне відділення має певну адресу, яка потім враховується під час найму працівників, місцезнаходження замовлень, а також для інших бізнес-процесів. Таблиця PLACE має чотири поля:

– ID INT(11) – ідентифікаційний номер;
– ADDRESS VARCHAR(255) – повна адреса відділення сервісного центру.

Таблиця REQUEST використовується для зберігання інформації про замовлення, має чотири поля:

– ID INT(11) – ідентифікаційний номер;
– DATE DATE – дата створення замовлення;
– DESCRIPTION VARCHAR(45) – опис замовлення;
– CLIENT_ID INT(11) – посилання на клієнта, який зробив замовлення.

Кожне замовлення змінює статуси залежно від того, на якому етапі просування воно знаходиться. Усі статуси зберігаються у спеціальній таблиці

– STATUS:
– ID INT(11) – ідентифікаційний номер;
– STATUS VARCHAR(255) – назва статусу;

Таблиці REQUEST і STATUS мають зв'язок багато до багатьох. Суміжною таблицею є таблиця STATUS_HISTORY. У цій таблиці зберігається:

– DATE DATE – дата додавання статусу до замовлення;
– STATUS_ID INT(11) – посилання на статус;
– REQUEST _ID INT(11) – посилання на замовлення.

У таблиці DEFECT зберігаються несправності техніки, має лише два поля:

– ID INT(11) – ідентифікаційний номер;
– NAME VARCHAR(45) – назва несправності.

Так як кожне замовлення може мати кілька несправностей, причому одна і та ж несправність може бути у різних замовлень, то таблиця DEFECT пов'язана відношенням багато до багатьох з таблицею REQUEST.

Відомості про клієнта знаходяться у таблиці CLIENT:

– ID INT(11) – ідентифікаційний номер;
– NAME VARCHAR(45) – ім'я користувача;
– SECOND_NAME VARCHAR(45) – по батькові;
– FAMILY_NAME VARCHAR(45) – прізвище;
– ADDRESS VARCHAR(45) – адреса проживання (використовується при доставці замовлення додому);
– PHONE VARCHAR(45) – номер телефону для зв'язку.

У процесі ремонту замовлення пересувається від співробітника до співробітника і виникає потреба зберігати інформацію про те, у якого працівника зараз знаходиться замовлення. Для цього таблиця REQUEST і USERS пов'язані ставленням багатьох до багатьох за допомогою таблиці REQUEST_USERS:

– ID INT(11) – ідентифікаційний номер;
– DATE DATE – дата;
– REQUEST _ID INT(11) – посилання на замовлення;
– USER_ID INT(11) – посилання на користувача системи.

Структурну схему бази даних наведено у Додатку А.

4.3 Розробка рівня взаємодії з базою даних

Оскільки всі дані програми зберігаються у базі даних, необхідно налаштувати взаємодію СУБД і розроблюваної системою. Мова Java є об'єктно-орієнтованою мовою програмування, тому всі дані в ній повинні бути представлені у вигляді об'єктів. Фреймворк Hibernate надає можливість об'єктно-реляційного відображення даних. У базі даних зараз знаходиться дванадцять сутностей. Всі ці сутності необхідні для роботи з користувачем, тому необхідно створити для кожної сутності об'єкт, який відображав би структуру сутності в базі даних. У структурі об'єкта необхідно вказати тільки ті атрибути з бази даних, які необхідні для роботи програми, таким чином скорочується необхідний додатком обсяг оперативної пам'яті.

Для того щоб використовувати вибрану реалізацію JPA фреймворк Hibernate необхідно прописати залежність у файлі pom.xml та заново імпортувати всі бібліотеки. Maven автоматично підтягне всі бібліотеки. Для роботи з цим фреймворком необхідно вказати адресу, за якою розташована база даних, назва схеми, ім'я користувача та пароль у базі даних.

Над створеним класом сутності необхідно вказати дві анотації @Entity та @Table в атрибутах останньої необхідно вказати назву сутності у базі даних. Замість анотацій також можна використовувати XML конфігурацію. Над кожним полем класу вказується анотація @Column, в атрибутах якої вказується назва атрибуту сутності базі даних, і навіть додаткові параметри. Як атрибут Java класу також може бути підмножина інших сутностей. Підмножини сутностей у Java класі є колекцією даних Set. Використання цього способу зберігання обумовлено тим, що при такому зберіганні даних відсутнє дублювання об'єктів. У Java класі над полем такого підмножини прописується інструкція @OneToMany, в атрибутах цієї інструкції вказується метод вилучення даних. Існує два способи EAGER та LAZY. Використання другого способу більш переважно, так як у порівнянні з EAGER дані вилучаються з бази даних тільки під час виклику get

методу даного атрибута, при використанні EAGER витягуються відразу всі дані з сутності, що відповідає даному Java об'єкту. Відповідно до специфікації в Java класі, що відображає сутність у базі даних, необхідно створити не приватний конструктор без аргументів.

Для операцій з сутностями використовується спеціальний інтерфейс JPA EntityManager. Даний інтерфейс дозволяє проводити основні операції над сутностями такі як додавання, зміна, пошук, видалення та блокування для зміни від інших потоків. EntityManager визначає лише один сеанс взаємодії з базою даних. Всередині себе EntityManager містить інтерфейс EntityTransaction, який дозволяє створювати транзакції, застосовувати зміни в базі даних і відкочувати ці зміни назад. Використання транзакцій дозволяє проводити безпечні зміни в базі даних. Конкретна реалізація EntityManager визначається класом EntityManagerFactory - це фабрика з'єднань, що підтримує з'єднання з базою даних, всю конфігурацію та кеш для роботи з базою даних основне завдання EntityManagerFactory створювати об'єкти інтерфейсу EntityManager [17].

Процес додавання об'єкта до бази даних полягає у створенні конкретного екземпляра класу, заповнення його атрибутів даними, валідації даних на правильність заповнення. Після цього за допомогою інтерфейсу EntityManager та його методів persist та merge.

На рисунку 4.1 представлена загальна схема взаємодії DataAccess шару з базою даних з використанням зв'язки JPA та Hibernate.

В даному проекті на даний момент створено 12 класів, що відображають сутності в базі даних. Для кожної сутності у шарі роботи з базою даних створено спеціальний клас, який відповідає за операції вставки, видалення, оновлення та пошуку даних.

У якості патерну використання конкретного DataAccess сервісу застосовуються Singleton. Це дозволяє використовувати один екземпляр класу в роботі всієї програми. Досягається це використанням Фреймворку Spring та його можливості управлінням впровадження залежностей Dependency Injection (DI).

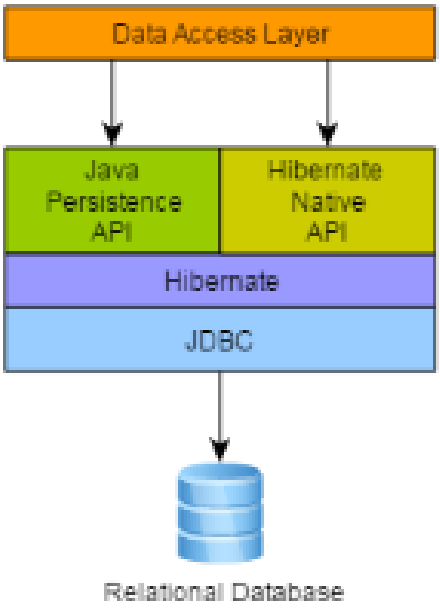

Рисунок 4.1 – Загальна схема взаємодії шару DataAccess з базою даних

У фреймворку Spring існує кілька способів управління використанням залежностей: XML конфігурація, інструкції, використання спеціальних інтерфейсів фреймворку Spring для зміни залежностей за допомогою Java коду. У цьому проекті використовуються перші два способи. У конфігурації XML вказується конкретна реалізація інтерфейсу EntityManagerFactory і JpaTransactionManager. Розроблені послуги DataAccess шару конфігуруються лише за допомогою анотацій. Анотація @Repository показує, що цей клас функціонує як репозиторій, що вимагає прозорої трансляції винятків. Незалежно від використовуваної технології у шарі доступу даних шар сервісу матиме справу із загальною ієрархією винятків Spring (DataAccessException). Над методами класів безпосередньо здійснюють роботу з базою даних стоїть інструкція @Transactional перед виконанням даного методу Spring відкриває транзакцію, а після виконання методу транзакція комітиться, а при викиданні RuntimeException відбувається відкат транзакції.

4.5 Розробка рівня бізнес-сервісів

У цьому проекті є велика бізнес логіка, щоб якось відокремити бізнес процеси від технічних сервісів системи, вся логіка пов'язана з бізнес процесами винесена в окремий шар званий Business Service (BS).

Під час створення даного шару активно використовується фреймворк Spring та її можливість використання залежностей.

У даному проекті, як вже було описано вище, для кожної сутності наявної в базі даних є її об'єкт, що відображається в Java коді. За аналогією з об'єктами званими моделями, були створені послуги, що дозволяють здійснювати роботу з базою даних.

У проекті на даний момент створено 6 сервісів. ClientService відповідає за всі операції, що проводяться з клієнтами: додати нового клієнта, знайти клієнта на прізвище, змінити дані клієнта. Кожен сервіс містить у собі операції з моделями. Такими операціями, як створення замовлення, зміна замовлення, видалення.

Наприклад, при створенні нового замовлення частина полів для вставки в базу даних автоматично заповнюється саме в шарі бізнес сервісу. Статус замовлення, місце його реєстрації та фактичного знаходження, а також відповідальний працівник встановлюються автоматично за рахунок повторного використання сервісів. Такий підхід дозволяє використати окремі бізнес-процеси повторно.

Об'єкти в додатку залежать один від одного. Хоча платформа Java забезпечує безліч функціональних можливостей розробки додатків, їй не вистачає засобів для організації базових будівельних блоків у єдине ціле, залишаючи це завдання архітекторам та розробникам. Для вирішення цієї проблеми можна використовувати шаблони проектування, такі як Factory, Abstract Factory, Builder, Decorator та Service Locator для складання різних класів та екземплярів об'єктів, що становлять додаток, ці шаблони містять: кращі практики із зазначенням імені,

з описом що робить шаблон, де його застосовувати, проблеми, з якими він поводиться, і так далі. Шаблони - це формалізовані найкращі практики, які можна реалізувати у своєму додатку. Компонент Spring Framework Inversion of Control (IoC) вирішує цю проблему, надаючи формалізовані засоби компонування розрізнених компонентів, готових до використання. Spring Framework кодує формалізовані шаблони проектування як об'єкти першого класу, які можна інтегрувати у власні програми.

Управління залежностями та впровадження залежностей – це різні речі. Щоб отримати ці приємні функції Spring у нашому додатку, необхідно зібрати всі необхідні бібліотеки (файли jar) і отримати їх на шляху до класів під час виконання і, можливо, під час компіляції. Ці залежності не є імпортованими віртуальними компонентами, а фізичними ресурсами у файловій системі. Процес управління залежностями включає розміщення цих ресурсів, їх зберігання і додавання їх у шляху до класів. Залежності можуть бути прямими (наприклад, моя програма залежить від Spring під час виконання) або непрямим (наприклад, моя програма залежить від commons-dbcp, який залежить від загального пулу). Непрямі залежності також відомі як «транзитивні», і саме ці залежності складніше ідентифікувати та керувати.

У кожен бізнес-сервіс за допомогою фреймворку Spring впроваджується залежність від сервісу, що відповідає за взаємодію з базою даних. З допомогою цієї залежності бізнес послуги працюють із даними що знаходяться у базі даних.

4.2 Огляд інтерфейсу та функціоналу застосунку

Завдяки Фреймворку ZK інтерфейс користувача розробляється досить швидко. Оскільки до програми є доступ з мережі Інтернет, необхідно продумати безпечне використання програми користувачами. Доступ користувачів до системи здійснюється за допомогою авторизації та аутентифікації. "Автентифікація" - це процес встановлення принципала, це означає, що користувач, пристрій або будь-

яка інша система може виконувати дію у додатку. "Авторизація" відноситься до процесу прийняття рішення про те, чи дозволено учаснику робити дії у вашому додатку. У розділі проектування цих цілей було прийнято рішення використовувати Spring Security.

Spring Security забезпечує комплексне рішення для забезпечення безпеки для корпоративного програмного забезпечення на базі Java EE програми. Особлива увага приділяється підтримці проектів, створених з використанням Spring Framework, який є провідним рішенням Java EE для розробки корпоративного програмного забезпечення.

Більшість причин для залучення Spring Security до проекту є виявлення функцій безпеки специфікації сервлетів Java EE або специфікації EJB, оскільки не вистачає глибини, необхідної для типових сценаріїв корпоративних додатків. Незважаючи на згадування цих стандартів, важливо визнати, що вони не переносяться на рівні WAR або EAR. Тому, якщо робота програми переключається на різні серверні середовища, зазвичай потрібна велика робота з переналаштування безпеки програми на нове цільове середовище. Використання Spring Security дозволяє подолати ці проблеми, а також приносить десятки інших корисних функцій безпеки, що настроюються.

Spring Security має архітектуру, яка призначена для поділу аутентифікації від авторизації та має стратегії та точки розширення для обох. При відкритті сайту користувачу необхідно пройти авторизацію (рис. 4.2).

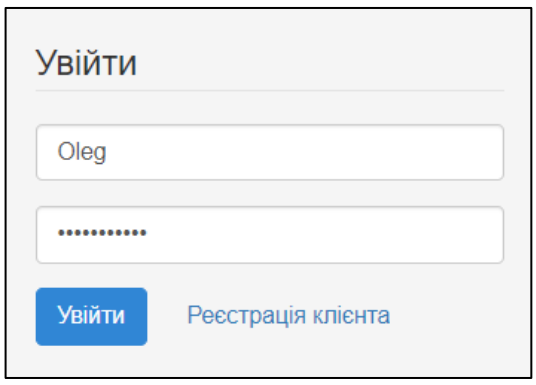

Рисунок 4.2 – Форма авторизації

У залежності від введених даних відбувається авторизація користувача з відповідною роллю. Загалом у системі їх існує 4 – директор (адміністратор), майстер, менеджер з продажів та клієнт. Для зручності розробки меню сайту було прийнято рішення використати відразу два шаблони взаємодії клієнта та сервера MVVM та MVC. Шаблон MVVM використовується для заповнення компонента даними, а шаблон MVC для керування поведінкою компонентів меню.

При натисканні на нижню кнопку меню сайту згортається ліворуч. Механізм згортання здійснюється шляхом зміни CSS стилю Java коді.

Для створення даного компонента сайту необхідно створити файл з розширенням. zul, у ньому за допомогою спеціальної мови описати, як має виглядати компонент. Також для форматування відображення компонента на сторінці необхідно прописати CSS стилі компонента

Приклад панелі адміністратора із можливістю додавання та видалення працівників наведено на рисунку 4.3.

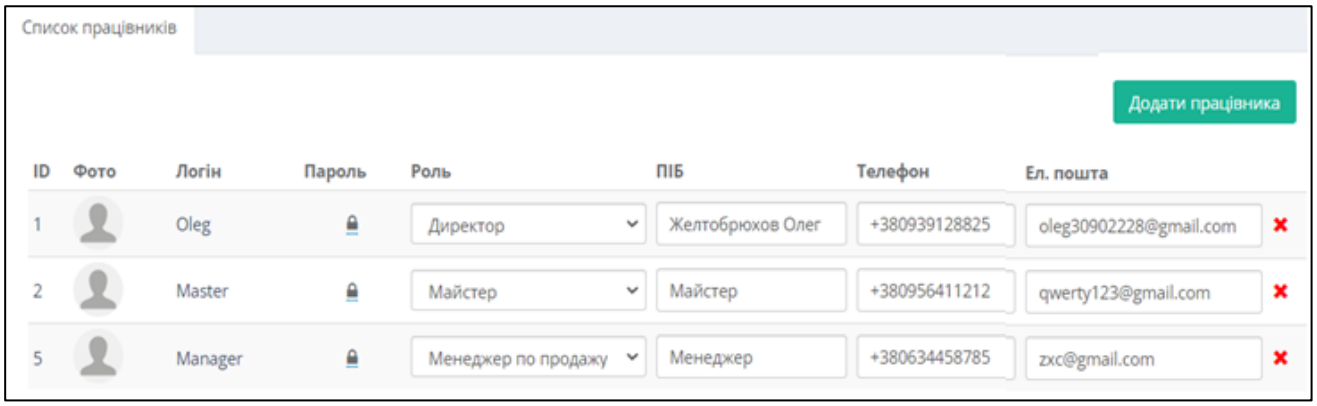

Рисунок 4.3 – Список працівників та їх дані

Для створення меню у фреймворку ZK є спеціальний тег «navbar», за допомогою цього тегу можна легко та швидко створити меню для сайту. У цьому компоненті можна налаштувати спосіб відображення вказавши в атрибуті "orien" значення "vertical" або "horizontal". У кожному компоненті, що надається

фреймворком ZK, можна вказати CSS стиль відображення. Тег "navitem" оголошує пункт меню, всередині нього необхідно вказати назву пункту, яка буде відображатися користувачеві в атрибуті "label". Є два основні способи завдання значення для рядкових констант на сайті. Перший спосіб полягають у винесенні всіх значень у спеціальний файл з розширенням .properties, в даному файлі оголошуються константи та їх значення. Значення атрибута «label» встановлюють так: «${labels.*}», де замість зірочки вказується константа, значення якої необхідно для відображення користувач. Цей підхід використовується для створення сайтів з можливістю змін мови інтерфейсу, також цей підхід дозволяє використовувати одну константу в різних частинах інтерфейсу. Другий спосіб полягає у вказівці значення атрибута "label" безпосередньо, прописуючи в лапках, що вивести користувачеві. Оскільки сайт не передбачає зміну мови інтерфейсу користувача, при розробці використовується другий спосіб. Для пункту меню замовлення вказано значення атрибуту selected="true", тому що це головна сторінка сайту, і при відкритті сторінки сайту користувачеві має відображатися ця сторінка. Для обробки подій на формі необхідний механізм, який зв'язує команди користувачів та обробки цих подій на сервері.

Команда – це дія для маніпулювання якістю ViewModel. Кожна команда позначає метод, який View може виконувати ViewModel. Ці дії також дозволяють користувачам взаємодіяти з переглядом. Наприклад, ViewModel надає 2 команди: "зберегти" та "видалити". Це означає, що користувачі можуть виконувати лише ці 2 дії у поданні за допомогою ViewModel.

Оскільки ViewModel діє як контролер, існує можливість зв'язати подію компонента інтерфейсу користувача з командою, вказавши ім'я команди, яке аналогічно зареєструвати в слухачі подій. Кілька подій можуть зв'язуватися з однією командою. Коли користувач взаємодіє з компонентом (наприклад, натисканням кнопки) компонент запускає подію, механізм прив'язки даних запускає виконання команди. Команда реалізується як спосіб ViewModel. Оскільки ViewModel є звичайним Java об'єктом, щоб механізм прив'язки даних

ідентифікував, який метод є командою, необхідно анотувати метод за допомогою ZK за допомогою анотації @Command. Дані методи керують властивістю ViewModel, наприклад, видаленням елемента.

Команда – це дія для маніпулювання якістю ViewModel. Кожна команда забезпечує дію, яку вигляд може виконувати в ViewModel. Необов'язковий елемент анотації - це ім'я рядка імені команди, і це ім'я посилається на ZUL з прив'язкою до подій. Якщо його не вказано, ім'я методу встановлюється як ім'я команди за промовчанням.

Команда ViewModel це як обробник подій, до кожної події прив'язана команда на сервері. Зв'язування між подіями та командою називається «прив'язкою Event-Command». Перш ніж встановити це зобов'язання, ми повинні оголосити команду з її ім'ям у ViewModel. Імена команд ViewModel не повинні дублюватися або це призведе до виключення під час виконання.

У разі авторизації на сайті сервіс-центру по обслуговуванню мобільних пристроїв користувача з роллю «Клієнт» відкривається форма заповнення заявки (рис. 4.4). Список зареєстрованих клієнтів видно адміністратору так, як показано на рисунку 4.5.

Клієнт має заповнити усі поля щоб залишити заявку на ремонт та подальший огляд її майстром, такі як особистий номер телефону (якщо він не збігається з номером, вказаним при реєстрації), ПІБ, інформацію щодо мобільного пристрою, який необхідно полагодити, опис несправності та наявність зовнішніх дефектів, які присутні на мобільному пристрої до здачу у ремонт.

Підтвердження введення користувача є незамінною функцією веб-програми. Валідатор ZK допомагає виконати це завдання. Валідатор є елементом багаторазового використання, який виконує перевірку та зберігає повідомлення перевірки у списку повідомлень перевірки. Коли він застосовується, він викликається перед збереженням даних у цільовій прив'язці (ViewModel або проміжний об'єкт). Якщо перевірка не виконана, властивості ViewModel (або проміжного об'єкта) не буде змінено.

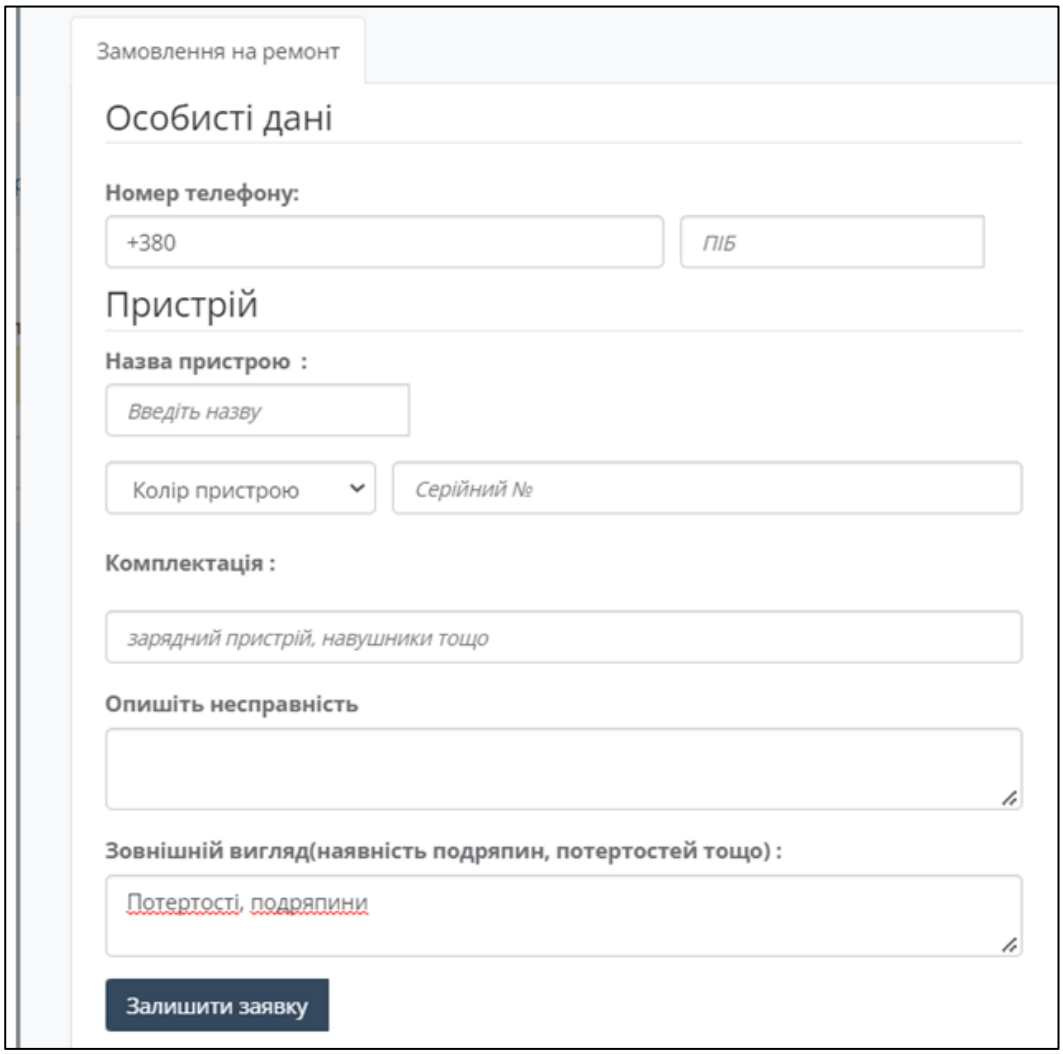

Рисунок 4.4 – Форма подання заявки клієнтом до сервіс-центру на ремонт мобільного пристрою

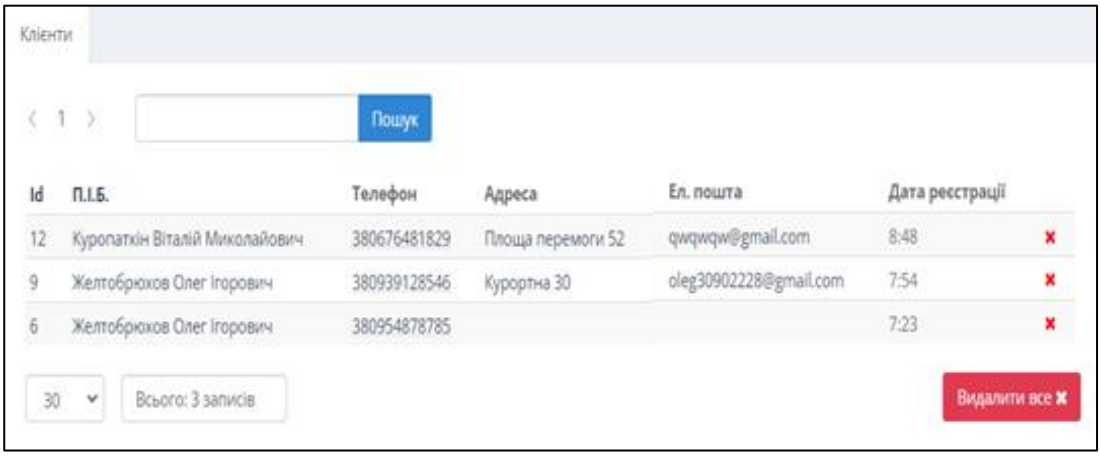

Рисунок 4.5 – Список зареєстрованих клієнтів

Databinding забезпечує стандартний механізм зберігання та відображення повідомлень перевірки. Після перевірки валідатор може зберегти повідомлення перевірки у списку повідомлень підтвердження. Приклад заповнення форми показано на рисунку 4.6.

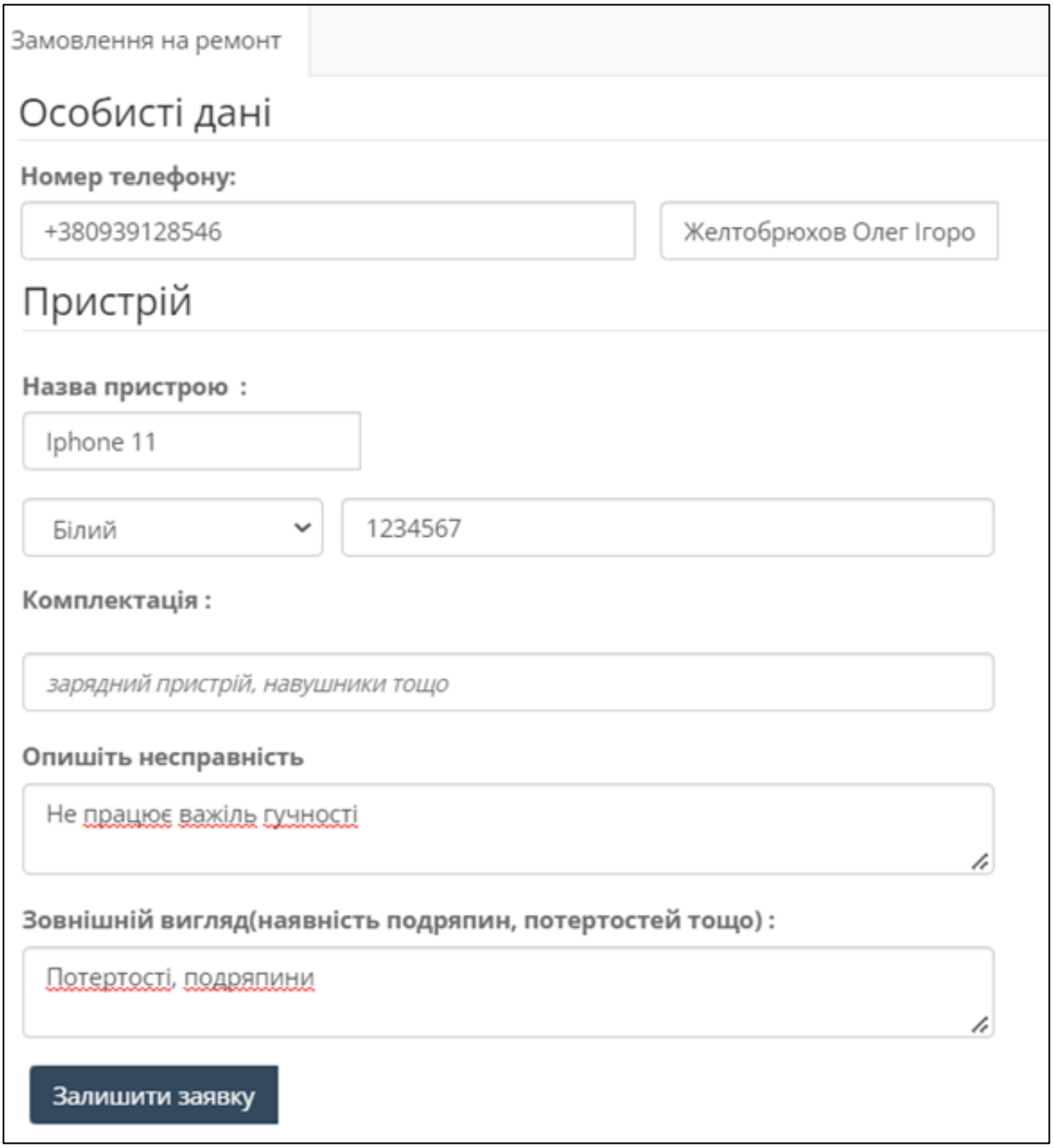

Рисунок 4.6 – Заповнена клієнтом форма заявки

Пункт меню замовлення містить інформацію про всі замовлення на ремонт у системі. Майстер на цій сторінці може переглядати замовлення, додавати та змінювати їх. Для відображення даних списковим способом фреймворк ZK надає готовий компонент listbox, що підтримує посторінкове представлення даних користувачеві, поділ даних на стовпці і рядки. Переміщення по списку можна здійснювати за допомогою клавіш. Приклад огляду заявок показано на рисунку 4.7.

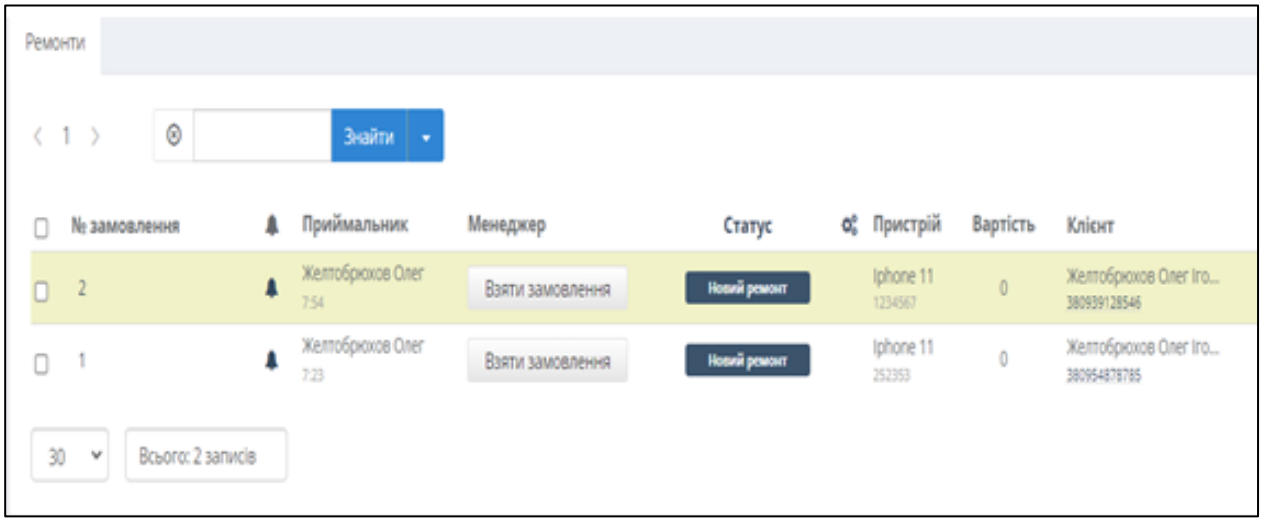

Рисунок 4.7 – Список поточних заявок у сервіс-центрі з обслуговування мобільних пристроїв

Для створення даного компонента потрібно за аналогією з компонентом меню, потрібно створити файл з розширенням.zul і прописати необхідні теги та їх атрибути.

Зв'язування даних на сервері та браузері клієнта подібно до буфера. Він автоматично створює об'єкт посередника. Перед збереженням у ViewModel усі вхідні дані зберігаються на об'єкті посередника. Таким чином, ми можемо зберігати дані від збереження в ViewModel до підтвердження користувача. Припускаючи, що користувач заповнює форму у веб-додатку, дані введення користувача безпосередньо зберігаються у властивостях ViewModel, цільового об'єкта. Потім користувач може скасувати дію заповнення перед відправкою

форми, тому дані, що зберігаються в ViewModel, застаріли. Це може викликати проблеми, якщо ми опрацюємо застарілі дані далі, тому вхідні дані спочатку зберігаються в об'єкті посередника, перш ніж перейти до реального цільового об'єкта після підтвердження користувача. Зв'язування форм забезпечує зберігання в об'єкті посередника інформації, що не підтверджена користувачем. Зв'язування форми дозволяє зберегти цільовий об'єкт у ViewModel без змін до виконання команди для підтвердження. Перед збереженням властивостей ViewModel (цільового об'єкта) за допомогою команди, ми можемо зберегти введення в об'єкті посередника. Коли команда виконується (наприклад, натиснута кнопка), вхідні дані дійсно зберігаються у властивостях ViewModel. Фреймворк дозволяє легко досягти даного ефекту, просто написавши вираз ZK bind, оскільки він знижує навантаження розробника на очищення неправильних даних вручну або реалізацію буфера. На рисунку 4.8 зображено потік даних між ZUL, об’єктом посередником і цільовим об’єктом.

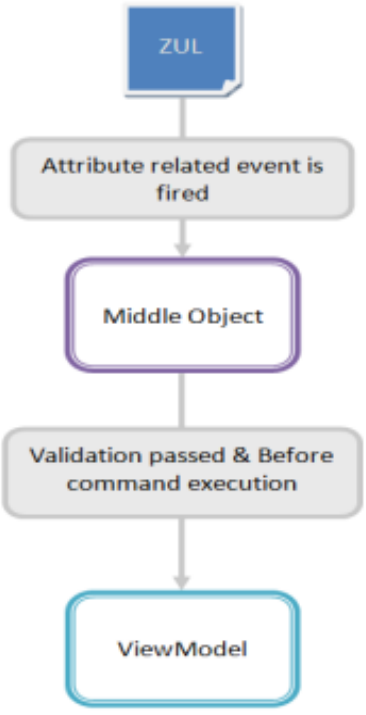

Рисунок 4.8 – Потік даних між ZUL, об’єктом посередником і цільовим об’єктом.

Для обробки заявки необхідно натиснути на її номер замовлення. Після цього відкривається форма, в якій майстру видно повну інформацію про клієнта та пристрій, що підлягає ремонту. На рисунку 4.9 наведено наглядний приклад заповнення частини заявки з приводу ремонту, де вводяться необхідні для ремонту деталі та вартість роботи майстра.

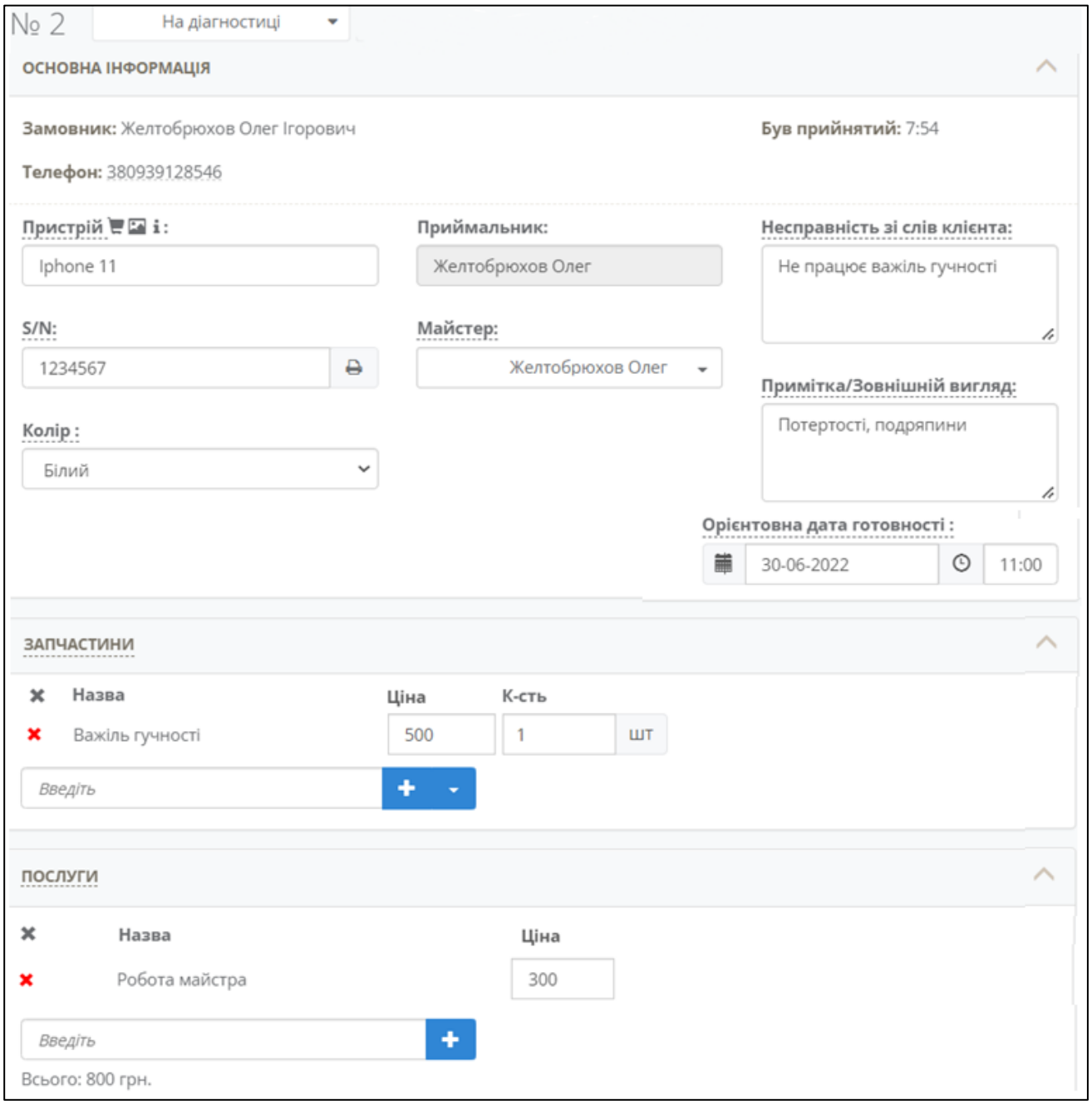

Рисунок 4.9 – Список поточних заявок у сервіс-центрі з обслуговування мобільних пристроїв

Після обробки заявки зміни зберігаються у БД та відображаються у формі перегляду усіх активних заявок. Статус замовлення змінюється на «На діагностиці» та доповнюється уся необхідна інформація так, як показано на рисунку 4.10.

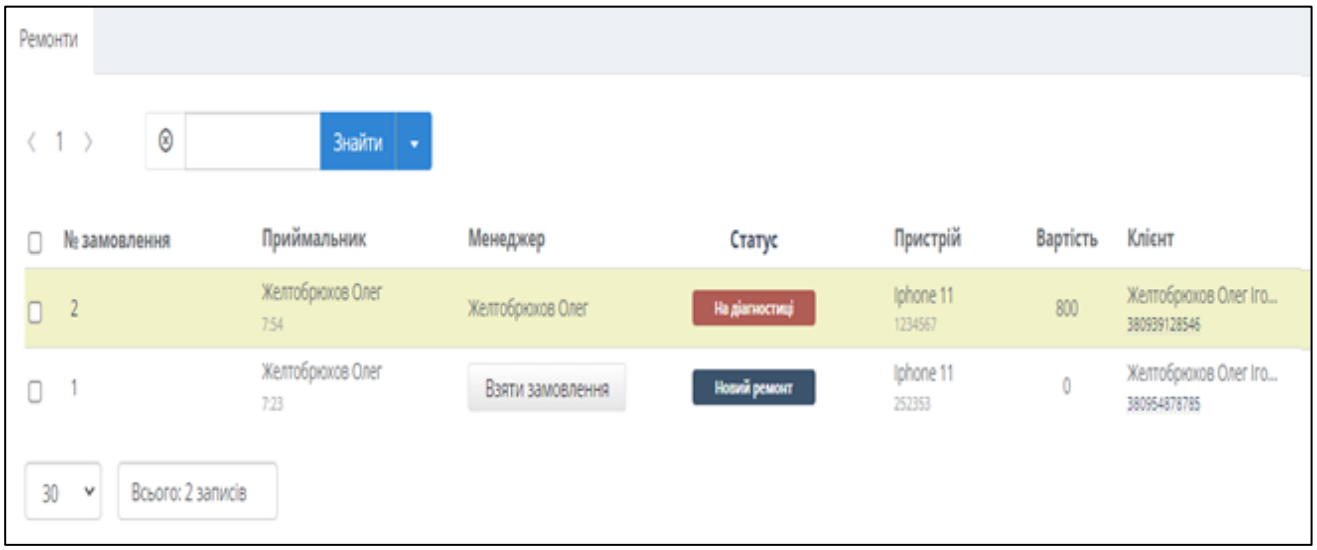

Рисунок 4.10 – Зміни у формі перегляду активних замовлень після обробки заявки

Загалом у застосунку реалізовано 7 статусів замовлення, а саме:

- Новий ремонт (при подачі заявки)
- На діагностиці (замовлення оброблено майстром)
- На узгодженні з клієнтом (коли вартість та орієнтовні строки ремонту обговорюються з замовником)
- В процесі ремонту
- Готовий (ремонт завершено та очікує на видачу клієнту після оплати)
- Видане
- Не підлягає ремонту (у разі відхилення заявки)

Форма зміни статусу наведено на рисунку 4.11.

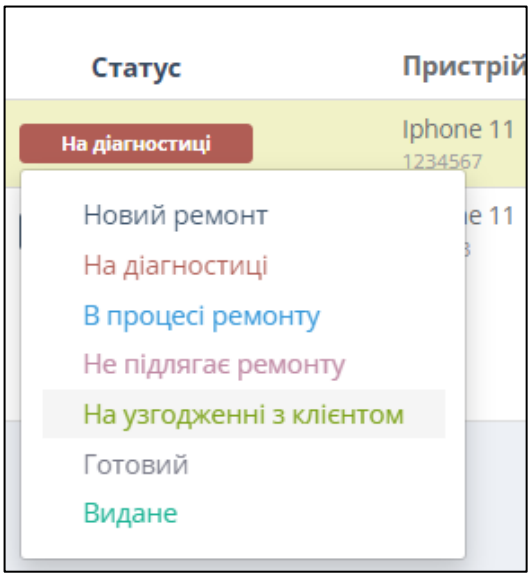

Рисунок 4.11 – Статуси замовлення протягом ремонту

4.3 Тестування застосунку

Тестування програмного забезпечення – це розслідування, яке проводиться з метою надання зацікавленим сторонам інформації про якість продукту або послуги, що тестується. Тестування програмного забезпечення також може забезпечити об'єктивне незалежне представлення програмного забезпечення, що дозволяє бізнесу оцінити та зрозуміти ризики реалізації програмного забезпечення. Методи тестування включають процес виконання програми або програми з метою пошуку помилок програмного забезпечення (помилок або інших дефектів) і перевірки того, що програмний продукт підходить для використання.

Оскільки кількість можливих тестів навіть для простих програмних компонентів практично нескінченна, все тестування програмного забезпечення використовує деяку стратегію для вибору тестів, які можливі для доступного часу і ресурсів. В результаті тестування програмного забезпечення зазвичай (але не виключно) намагається виконати програму або програму з метою пошуку помилок програмного забезпечення (помилок чи інших дефектів). Робота з тестування – це ітеративний процес, коли фіксується одна помилка, вона може

висвітлювати інші, глибші помилки чи навіть створювати нові. Тестування програмного забезпечення може забезпечити об'єктивну, незалежну інформацію про якість програмного забезпечення та ризик його відмови користувачам чи спонсорам.

Тестування програмного забезпечення може проводитися, як тільки програмне забезпечення (навіть якщо воно частково завершено) існує. Загальний підхід до розробки програмного забезпечення часто визначає коли і як проводиться тестування. Наприклад, у поетапному процесі більшість тестів відбувається після того, як вимоги до системи були визначені, а потім реалізовані в програмах, що тестуються. Навпаки, під Agile підхід, вимоги, програмування та тестування часто виконуються одночасно.

Під методиками мається на увазі створення плану, як реалізувати ідею, а техніка – метод чи спосіб виконання завдання. Таким чином, методики тестування створюють набір входів даного програмного забезпечення, яке забезпечить набір очікуваних результатів. Ідея полягає в тому, щоб система працювала досить добре, і вона може бути випущена з мінімальними проблемами кінцевого користувача.

Статичні методи випробувань забезпечують відмінний спосіб покращити якість та продуктивність розробки програмного забезпечення. Він включає огляди та дає огляд того, як вони проводяться. Основне завдання статичного тестування – покращити якість програмних продуктів, допомагаючи інженерам розпізнавати та виправляти власні дефекти на ранній стадії процесу розробки програмного забезпечення.

При тестуванні програмного продукту застосовуються методики системного тестування. Тестування системи - це метод тестування чорної скриньки, виконаний з метою оцінки всієї системи відповідності системи заданим вимогам. При тестуванні системи функціональність системи тестується з погляду наскрізної перспективи. Системне тестування зазвичай виконується командою, яка залежить від команди розробників, щоб виміряти якість системи неупереджено. Він включає як функціональне, так і не функціональне тестування.

Тестування програми, що розробляється, проводиться шляхом тестування функцій сайту, таких як додавання клієнта, додавання замовлення, перегляд замовлення, зміна замовлення.

Під час тестування було перевірено всі функціональні елементи сайту. В результаті тестування всі функції сайту виконуються відповідно до вимог.

Оцінку відповідності функціональності програми до заявлених вимог наведено в таблиці 4.1.

Таблиця 4.1 – Параметри оцінки відповідності функціоналу програми

Функції програми	Реалізація	Виведення повідомлення у разі помилки
Реєстрація/ авторизація	Реалізовано під час входу користувача на сайт.	У разі введення помилкових даних виводиться повідомлення «Неправильне введення логіну та/або пароля»
Перегляд списків даних	Вибір пункту меню відповідного списку даних	У разі відсутності записів у списку виводиться повідомлення «Список порожній»
Не всі поля заповнені у формах створення	Створення нового користувача, замовлення, клієнта або будь-якого іншого запису	У разі заповнення не всіх обов'язкових полів виводиться помилка над кожним полем.
При неможливості підключення до БД	Надсилання запиту до бази даних	У разі неможливості підключення до БД помилка "Неможливо підключитися до БД"

Висновки до розділу 4

У четвертому розділі описано програмну реалізацію та детальний опис інтерфейсу розробленого web-застосунку. При розробці було використано шаблони взаємодії клієнта та сервера MVVM та MVC. Шаблон MVVM використовується для заповнення компонента даними, а шаблон MVC для керування поведінкою компонентів меню.

У результаті проведеного дослідження на основі розробленої моделі сервісного центру здійснено програмну реалізацію вебзастосунку інформаційної системи сервіс-центру по обслуговуванню мобільних пристроїв, яка має зручний інтерфейс користувача, дозволяє у режимі реального часу відслідковувати стан замовлення, усуває неузгодженість у роботі продавців, маркетологів та служби підтримки клієнтів, автоматизує й полегшує діяльність працівників сервісного центру, підвищуючи ефективність їх роботи з прийому замовлень у центрі.

Було проведено тестування системи таких функцій, як додавання клієнта, додавання замовлення, перегляд замовлення, зміна замовлення. Під час тестування було перевірено всі функціональні елементи вебзастосунку та виявлено, що всі функції сайту виконуються відповідно до вимог.

Спеціальний розділ

ОХОРОНА ПРАЦІ

до кваліфікаційної роботи

на тему:

ІНФОРМАЦІЙНА СИСТЕМА ПРИЙОМУ ЗАМОВЛЕНЬ У СЕРВІС-ЦЕНТРІ ПО ОБСЛУГОВУВААННЮ МОБІЛЬНИХ ПРИСТРОЇВ

Спеціальність 122 «Комп'ютерні науки»

122 – БКР – 401.21810110

Виконав студент 4-го курсу, групи 401
_____ О. І. Желтобрюхов
(підпис, ініціали та прізвище)
«___» червня 2022 р.

Консультант ст. викладач
_____ О.В. Макарова
(підпис, ініціали та прізвище)
«___» червня 2022 р.

Миколаїв – 2022

5 ОХОРОНА ПРАЦІ: САНІТАРНО-ГІГЄНІЧНІ ВИМОГИ ДО РОБОЧОГО МІСЦЯ КОРИСТУВАЧА ПК

З розвитком науково-технічного прогресу важливу роль грає можливість безпечного виконання людьми своїх трудових обов'язків. У зв'язку з цим була створена і розвивається наука про безпеку праці і життєдіяльності людини.

Безпека життєдіяльності (БЖД) – це комплекс заходів, спрямованих на забезпечення безпеки людини в середовищі проживання, збереження його здоров'я, розробку методів і засобів захисту шляхом зниження впливу шкідливих і небезпечних факторів до допустимих значень, вироблення заходів по обмеженню збитку в ліквідації наслідків надзвичайних ситуацій мирного і воєнного часу.

Охорона здоров'я працівників, забезпечення безпеки умов праці, ліквідація професійних захворювань і виробничого травматизму складає одну з головних турбот людського суспільства. Звертається увага на необхідність широкого застосування прогресивних форм наукової організації праці, зведення до мінімуму ручної, малокваліфікованої праці, створення обстановки, що виключає професійні захворювання і виробничий травматизм.

На робочому місці повинні бути передбачені заходи захисту від можливого впливу небезпечних і шкідливих факторів виробництва. Рівні цих факторів не повинні перевищувати граничних значень, обумовлених правовими, технічними і санітарно-технічними нормами. Ці нормативні документи зобов'язують до створення на робочому місці умов праці, при яких вплив небезпечних і шкідливих чинників на працівників або усунуто зовсім, або знаходиться в допустимих межах.

Враховуючи, що законодавством України проголошено пріоритет життя і здоров’я людини, **метою** розділу “Охорона праці” є створення безпечних і здорових умов праці на робочих місцях, в робочих зонах, у виробничих приміщеннях. Досягається це опрацюванням питань умов праці, гігієни праці і виробничої санітарії, техніки безпеки, пожежної безпеки, цивільного захисту,

екологічної безпеки та безпеки життєдіяльності людини в умовах надзвичайних ситуацій.

5.1 Організація охорони праці

Відповідно до ДСТУ 2293 - 93 "Охорона праці. Терміни та визначення" охорона праці - це система забезпечення життя і здоров'я працівника в процесі праці всіма способами і заходами: правовими, соціально-економічними, санітарними, гігієнічними, лікувальними та профілактичними, організаційно-технічними і ін.

Правовою основою законодавства про охорону праці є Конституція України, Закони України: "Про охорону праці", "Про охорону здоров'я", Кодекс законів про працю України, Закон України «Про загальнообов’язкове державне соціальне страхування від нещасного випадку на виробництві та професійного захворювання, які спричинили втрату працездатності», Закон України «Про забезпечення санітарного та епідемічного благополуччя населення», Положення про Державний комітет України з нагляду за охороною праці, Положення про Національну Раду з питань безпечної життєдіяльності населення й т. ін.

Основоположне значення в галузі охорони праці має Закон України «Про охорону праці», який визначає основні положення щодо реалізації конституційного права працівників на охорону їх життя і здоров’я у процесі трудової діяльності, на належні, безпечні і здорові умови праці, регулює за участю відповідних органів державної влади відносини між роботодавцем і працівником з питань безпеки, гігієни праці та виробничого середовища і встановлює єдиний порядок організації охорони праці в Україні.

Згідно з Законом України «Про охорону праці» служба охорони праці створюється власником або уповноваженим ним органом на підприємствах, в установах, організаціях незалежно від форм власності та видів їх діяльності для організації виконання правових, організаційно-технічних, санітарно-

гігієнічних, соціально-економічних і лікувально-профілактичних заходів, спрямованих на запобігання нещасним випадкам, професійним захворюванням і аваріям у процесі праці.

Відповідно до Типового положення служба охорони праці виконує такі основні ***функції***:

– проводить оперативно-методичне керівництво роботою з охорони праці;

– складає разом зі структурними підрозділами підприємства комплексні заходи щодо досягнення встановлених нормативів безпеки, гігієни праці та виробничого середовища;

– проводить для працівників увідний інструктаж з питань охорони праці;

– організовує забезпечення працюючих правилами, стандартами, нормами, положеннями, інструкціями та іншими нормативними актами з охорони праці;

– бере участь у розслідуванні нещасних випадків та аварій;

– веде облік, аналіз нещасних випадків, професійних захворювань і аварій, а також шкоди від цих подій і т. ін.

Загальний контроль за охороною праці здійснюється трудовими колективами (через обраних або уповноважених) і профспілками (в особі обраних органів і представників).

Охорона праці є обов'язковою при розробці різних пристроїв і механізмів.

5.1.1 Вплив комп’ютерної техніки на людину

Широке промислове та побутове використання ПК актуалізувало питання охорони праці їхніх користувачів. Найбільш повним нормативним документом щодо забезпечення охорони праці користувачів ПК є "Державні санітарні норми і правила роботи з візуальними дисплейними терміналами (ВДТ) електронно-обчислювальних машин" ДСанПіН 3.3.2.007-98.

Ігнорування санітарних правил і норм роботи з ВДТ може викликати у осіб, які з ними професійно працюють, загальну втому, зорову втому, болі та

відчуття піску в очах, відчуття засміченості та свербіння очей, болі в хребті, закам'янілість та оніміння м'язів шиї та плечового поясу, пошкодження дисків хребта, порушення постави, судоми м'язів ніг, синдром RSI хронічний розтяг зв'язок, синдром тунелю Карпаля, головні болі, поганий сон, депресивні стани.

5.1.2 Вплив шуму на людину

Шум як гігієнічний фактор - це сукупність звуків різної частоти й інтенсивності, які сприймаються органами слуху людини й викликають неприємне суб'єктивне відчуття.

Шум як фізичний фактор являє собою механічний коливальний рух, що хвилеподібно поширюється в пружному середовищі і носить звичайно випадковий характер. Він характеризується звуковим тиском як функцією частоти й часу.

З фізіологічної точки зору шум визначається як відчуття, що сприймається органами слуху під час дії на них звукових хвиль у діапазоні частот 16 - 20 000 Гц.

Наслідками шкідливої дії виробничого шуму можуть бути професійні захворювання, підвищення загальної захворюваності, зниження працездатності, підвищення ступеня ризику травм і нещасних випадків, пов'язаних з порушенням сприйняття попереджувальних сигналів, порушення слухового контролю функціонування технологічного встаткування, зниження продуктивності праці.

Шум електромагнітного походження виникає внаслідок коливань елементів електромеханічних пристроїв (ротора, статора, сердечника, трансформатора й т.д.) під впливом змінних магнітних полів.

Тиск, що перевищує атмосферний, називається акустичним, або звуковим тиском. Чим більший звуковий тиск, тим голосніше звук.

Мірою інтенсивності звукових хвиль у будь-якій точці простору є величина звукового тиску - надлишковий тиск у даній точці середовища в

порівнянні з тиском при відсутності звукового поля. Одиниця виміру звукового тиску р - Н/м^2 (1 Н/м^2 = 1 Па).

По характері порушення фізіологічних функцій шум розділяється на:

– такий, що заважає (перешкоджає язиковому зв'язку);

– дратівний (викликає нервова напруга й внаслідок цього - зниження працездатності, загальна перевтома);

– шкідливий (порушує фізіологічні функції на тривалий період і викликає розвиток хронічних захворювань, які безпосередньо зв'язані зі слуховим сприйняттям: погіршення слуху, гіпертонія, туберкульоз, виразка шлунку);

– такий, що травмує (різко порушує фізіологічні функції організму людини).

Шум, навіть коли він невеликий (при рівні 50-60 дБ), створює значне навантаження на нервову систему людини, роблячи на нього психологічний вплив. Відсутність необхідної тиші, особливо в нічний час, приводить до передчасної утоми, а часто й до захворювань. У цьому зв'язку необхідно відзначити, що шум в 30-40 дБ в нічний час може бути серйозним фактором, що турбує.

Впливаючи на кору головного мозку, шум робить дратівну дію, прискорює процес стомлення, послабляє увагу й сповільнює психічні реакції.

Таким чином, шум викликає небажану реакцію всього організму людини. Патологічні зміни, що виникли під впливом шуму, розглядають як шумову хворобу.

5.1.3 Недостатня освітленість робочого місця

Напружена зорова робота в результаті нераціонального освітлення може бути причиною функціональних порушень у зоровому аналізаторі й привести до розладу зору, а у важких випадках - і до повної втрати.

Утома органів зору залежить від ступеня напруженості процесів, які супроводжують зорове сприйняття.

Залежно від джерела світла виробниче освітлення може бути трьох видів: природним, штучним і з'єднаним.

Штучне освітлення в приміщеннях з робочим місцем, обладнаним ВДТ, має здійснюватись системою загального рівномірного освітлення. Як джерело штучного освітлення мають застосовуватись люмінесцентні лампи ЛБ.

5.2 Заходи електробезпеки

Відповідно до вимог для попередження ураження електричним струмом необхідно:

– чітко й у повному обсягу виконувати правила виконання робіт і правил технічної експлуатації;

– виключити можливість доступу адміністратора до частин устаткування, що працює під небезпечною напругою, неізольованим частинам, призначеним для роботи при малій напрузі і непідключеним до захисного заземлення;

– застосовувати ізоляцію, що служить для захисту від ураження електричним струмом, виконану з застосуванням міцного суцільного чи багатошарового ізоляційного матеріалу, товщина якого обумовлена типом забезпечуваного захисту;

– підводити електроживлення до ЕОМ від розетки будинку за допомогою спеціальної вилки з контактом, що заземлено;

– захистити від перевантажень по струму, розраховуючи на потужність, що споживається від мережі; а також захистити від короткого замикання устаткування, вбудованого в мережу будинку;

– надійно підключити до затисків, що заземлюють, металеві частини, доступні для адміністратора, що у результаті ушкодження ізоляції можуть виявитися під небезпечною напругою;

– перевірити, що захисний провідник, що заземлює, не має вимикачів і запобіжників, а також надійно ізольований.

У процесі роботи необхідно:

– не допускати до робочого місця осіб, що не мають відношення до виконуваної роботи;

– всі роботи по зміні модулів і блоків, заміні елементів проводити тільки в знеструмленому положенні;

– при вимірюванні параметрів, режимів роботи і параметрів електричних ланцюгів вживати всіх заходів обережності і уникати дотику до небезпечних ланцюгів;

– при заміні елементів і схем на місце вийшов з ладу ставити елемент відповідного номіналу;

– не допускається вживання елементів, розрахованих на інший струм або напругу;

– при спрацьовуванні системи захисту перед повторним включенням слід ретельно оглянути блоки споживача на предмет виявлення видимих пошкоджень монтажних і живлячих дротів.

При аваріях необхідно:

– негайно відключити живляче напругу;

– повідомити керівника робіт про аварію;

– якщо в результаті аварії постраждала людина, надати першу медичну допомогу;

– при виникненні пожежі вжити заходів по ліквідації вогнища наявними засобами, при необхідності викликати пожежну команду.

Пожежна безпека в обраному приміщенні забезпечується дотриманням вимог НПАОП 0.00-1.28-10 [2].

5.2.1 Захист від шкідливого впливу випромінювань та статичних зарядів

Для захисту від шкідливого впливу випромінювань можливе

застосування заземлених захисних екранів, що значно зменшують їхню інтенсивність. Крім того, рекомендується використовувати монітори, що відповідають специфікації MPR II, розробленої Шведською Національною Радою по Вимірах і Тестуванню (зазначено закордонний стандарт, тому що велика частина експлуатованої і придбаної обчислювальної техніки зроблена не в Україні). Специфікація визначає рівень електромагнітного випромінювання моніторів для двох смуг частот: 5 Гц – 2 кГц і 2 – 400 кГц.

Напруженість електричного поля в нижній смузі не повинна перевищувати 25 В/м, у верхньої – 2.5 В/м, відповідно напруженість магнітного поля 250 і 2.5 нТ.

Одним із варіантів зменшення величини статичних зарядів на технологічних об’єктах є використання полівінилхлоридного антистатичного покриття. Також статистичний заряд може бути нейтралізованим за допомогою іонізованого газу. Найбільш поширеними загальними заходами щодо зменшення величини статичного заряду є місцеве зволоження повітря.

5.2.2 Вимоги до освітлення приміщень та робочих місць

Освітленість на робочому місці повинна відповідати характеру зорової роботи, який визначається трьома параметрами: об'єктом розрізнення – найменшим розміром об'єкта, що розглядається на моніторі ПК; фоном, який характеризується коефіцієнтом відбиття; контрастом об'єкта і фону;

– необхідно забезпечити достатньо рівномірне розподілення яскравості на робочій поверхні монітора, а також в межах навколишнього простору;

– на робочій поверхні повинні бути відсутні різкі тіні;

– в полі зору не повинно бути відблисків (підвищеної яскравості поверхонь, які світяться та викликають осліплення);

– величина освітленості повинна бути постійною під час роботи;

– слід обирати оптимальну спрямованість світлового потоку і необхідний склад світла.

5.3 Розрахунок освітленості

Щоб визначити чи є освітленість достатньою, необхідно провести її розрахунок.

Нормований коефіцієнт природного освітлення для відповідної (заданої) категорії зорової роботі , %. Визначається відповідно до ДБН В.2.5-28-2006 [2]

Розрахунок освітленості робочого місця зводиться до вибору системи освітлення, визначенню необхідної кількості світильників, їхнього типу і розміщення. Виходячи з цього, розрахуємо параметри штучного освітлення.

Як правило, штучне освітлення здійснюється за допомогою електричних джерел світла двох видів: люмінесцентних ламп та ламп розжарювання. Розрахунки будуть здійснюватися для люмінесцентних ламп, оскільки у порівнянні з лампами розжарювання вони мають суттєві переваги:

- по спектральному складу світла вони найбільш близькі до природнього, денного світла;
- вищий ККД (у 1,5-2 рази вище, ніж ККД ламп розжарювання);
- більша світловіддача (у 3-4 рази вище, ніж у ламп розжарювання);
- більш довгий термін служби.

Розрахунок проводиться для кімнати з площею 11,25 м^2 , ширина якої 4,5 м, а довжина 2,5 м. Скористаємося методом світлового потоку.

Для визначення кількості світильників визначимо світловий потік, що падає на поверхню, за формулою:

$$F = \frac{E \cdot K \cdot S \cdot Z}{n}, \text{ де}$$

F – світловий потік, що розраховується, Лм;

E – нормована мінімальна освітленість, Лк. Робота з комп'ютером відноситься до розряду точних робіт, отже у нашому випадку ***E*** = 300 Лк;

S – площа приміщення. У нашому випадку 11,25 м^2;

Z – відношення середньої освітленості до мінімальної (як правило, це 1,1…1,2, нехай ***Z*** = 1,1);

K – коефіцієнт запасу, що враховує зменшення світлового потоку лампи у результаті забруднення світильників у процесі використання (його значення залежить від типу приміщення та характеру робот, що у ньому проводяться. У нашому випадку ***K*** = 1,5);

n – коефіцієнт використання (визначається відношенням світлового потоку, що падає на розрахункову поверхню, до сумарного потоку усіх ламп та обчислюється у частках одиниці; залежить від розмірів приміщення, характеристик світильника, забарвлення стелі та стін, що характеризуються коефіцієнтами віддзеркалення від стелі ($Рс_1$) та стін ($Рс_2$)). Значення коефіцієнтів були вказані вище: $Рс_1$ = 60%, $Рс_2$ = 40%. Значення ***n*** визначимо по таблиці коефіцієнтів використання різних світильників. Для цього обчислимо індекс приміщення по формулі:

$$I = \frac{S}{h \cdot (A+B)}, \text{ де}$$

S – площа приміщення, у нашому випадку ***S*** = 11,25 м^2;

h – розрахункова висота приміщення, ***h*** = 2,92 м;

A – ширина приміщення, ***A*** = 2,5 м;

B – довжина приміщення, ***B*** = 4,5 м.

Підставивши значення отримаємо такий розрахунок:

$$I = \frac{11{,}25}{2{,}92 \cdot (2{,}5 + 4{,}5)} = 0{,}55$$

Знаючи індекс приміщення ***I***, по таблиці коефіцієнтів використання лічильників з’ясовуємо, що ***n*** = 0,22.

Підставивши усі значення у формулу для визначення світлового потоку ***F***, отримаємо такий розрахунок:

$$F = \frac{300 \cdot 1{,}5 \cdot 11{,}25 \cdot 1{,}1}{0{,}22} = 25\,312{,}5 \text{ Лм}$$

Для освітлення вибираємо люмінесцентні лампи типу ЛБ40-1, світловий потік яких ***F*** = 4320 Лк.

Необхідну кількість таких ламп у приміщенні розрахуємо по формулі:

$$N = \frac{F}{Fл}, \text{ де}$$

N – кількість ламп, яку ми визначаємо;

F – світловий потік, ***F*** = 25 312,5 Лм;

Fл – світловий потік лампи, ***Fл*** = 4320 Лм.

Підставимо ці значення у формулу:

$$N = \frac{25\ 312{,}5}{4320} = 5 \text{ шт.}$$

Розрахована кількість люмінесцентних ламп типу ЛБ40-1 для освітлення приміщення площа якого складає 11,25 м^2. При використання штучного освітлення потрібно 5 таких ламп, при таких умова буде дотримано норми охорони праці щодо освітлення робочого місця.

Висновки до розділу 5

Одне з найбільш важливих завдань у розробці нових технологій і систем виробництва – вивчення й вирішення проблем, пов'язаних із забезпеченням здорових і безпечних умов, у яких відбувається праця людини. Дослідження й виявлення можливих причин виробничих нещасних випадків, професійних захворювань, аварій, вибухів, пожеж, і розробка заходів і вимог, спрямованих на усунення цих причин дозволяють створити безпечні й сприятливі умови для праці людини. Комфортні й безпечні умови праці − один з основних факторів, який впливає на продуктивність і безпеку праці, здоров’я працівників.

В даному розділі були описані вимоги до робочого місця користувача ПК. Створені умови сприяють комфортній та безпечній для здоров’я роботі. Проведено вибір системи і розрахунок оптимального освітлення виробничого приміщення на робочому місці. Виконано перевірочні розрахунки як природного так і штучного освітлення. Безумовно, покращення умов праці персоналу потребують певних матеріальних витрат, але в розвинутих країнах доведено, що вони повністю компенсуються за рахунок підвищення продуктивності праці.

ВИСНОВКИ

Підбиваючи підсумки можна сказати, що у будь-якій сфері професійної діяльності останнім часом вважається актуальним можливість спрощення обробки деяких даних. Одним із способів досягти цього є автоматизація цього процесу. Найважливішою з основних проблем у сервісних центрах є те, що вони приносять замало прибутку, тому, щоб збільшити ефективність їх роботи, варто розширити спектр послуг.

Здійснений аналіз показав, що сучасний стан інформатизації суспільства супроводжується розширенням сфери інтернет-сервісу для підтримки технічного обслуговування мобільних пристроїв. Установлено, що сервісний центр є організацією, яка займається наданням послуг із сервісної підтримки та обслуговування техніки, обладнання та іншої продукції. Діяльність сервісних центрів включає перед торговий, гарантійний та після продажний ремонт і постачання виробленої продукції. Інформаційна система для підтримки діяльності сервіс-центру є сукупністю програмних та апаратних засобів, а також організаційного забезпечення, які надають інформаційну підтримку клієнтам та працівникам під час обслуговування мобільних пристроїв.

Виявлено, що в Україні існує багато інтернет-ресурсів для автоматизації діяльності сервісних центрів: РемОнлайн, РемонтОнлайн, БазаКвитанций, WinService Pro, S-Center, MasterTool. Вони мають широкий функціонал, який дозволяє контролювати співробітників, вести складський та фінансовий облік, формувати та друкувати документи і звіти, забезпечувати комунікацію співробітників центру з клієнтами. Проте багато з них є комерційними та дорогими й не надають можливості для зміни функціоналу під конкретну організацію. Для вирішення цієї проблеми є доцільним розробка інформаційної системи сервісного центру прийому замовлень по обслуговуванню мобільних пристроїв, яка дозволить клієнтам оформляти замовлень у режимі онлайн.

Установлено, що одним із базових понять проєктування інформаційних систем є поняття життєвого циклу проєкту, який визначається моделлю й описується у формі методу – методології, яка визначає комплекс робіт, їх послідовність, детальний зміст та відповідальність фахівців на усіх етапах розробки ІС. Здійснено аналіз існуючих підходів до проектування інформаційної системи та виявлено найбільш поширені методології: SADT: методологія структурного аналізу, IDEF0: функціонального моделювання, DFD: діаграм потоків даних та каскадна модель. У результаті здійсненого аналізу обґрунтовано вибір методу побудови ІС на основі каскадної моделі та виявлено основні етапи цього методу: 1) формування вимог до ІС, 2) розробка концепції ІС, 3) розробка та затвердження ТЗ на створення ІС, 4) розробка ескізного проекту, 5) розробка технічного проекту, 6) розробка робочої документації, 7) впровадження розробленої ІС, 8) супровід ІС: гарантійне та післягарантійне обслуговування.

З’ясовано, що використання CMS для розробки інформаційної системи не є доцільним, оскільки всі CMS мають обмежений функціонал і не дозволяють повноцінно реалізувати складну бізнес логіку. Для розробки web-застосунку було обрано трирівневу архітектуру: 1) клієнтська частина: включає інтерфейс користувача і організацію взаємодії користувача з системою; 2) серверна частина: включає сервер, де реалізовано основну логіка програми: обробку даних, генерацію звітів, взаємодію з різними системами; 3) база даних: містить усю інформацію стосовно роботи системи.

Для розробки клієнтської частини web-застосунку було обрано HTML, CSS, JavaScript та фреймворк ZK, який дозволяє працювати з обраним шаблоном проєктування MVVM. Для розробки серверної частини обрано мову програмування Java, інтегроване середовище розробки IntelliJ Idea, пакет JDK та фреймворк з відкритим кодом Spring Framework. Для забезпечення безпеки обрано фреймворк Spring Security.

Для отримання й зберігання даних обрано базу даних MySQL, яка має зручний інструмент для розробки MySQL Workbench, що полегшує створення

схеми бази даних та виконання запитів до неї. Організацію взаємодії бази даних та програми на мові Java реалізовано з використанням бібліотеки Hibernate. Для автоматизованого виконання операцій компіляції, збирання, виконання тестів та додавання бібліотек до проєкту використано спеціалізований фреймворк збирач Apache Maven. При розробці було також вирішено також використовувати контейнер сервлетів та веб-сервер Apache Tomcat і систему контролю версій GIT.

У результаті проведеного дослідження на основі розробленої моделі сервісного центру здійснено програмну реалізацію та тестування вебзастосунку інформаційної системи сервіс-центру по обслуговуванню мобільних пристроїв, яка має зручний інтерфейс користувача, дозволяє у режимі реального часу відслідковувати стан замовлення, усуває неузгодженість у роботі продавців, маркетологів та служби підтримки клієнтів, автоматизує й полегшує діяльність працівників сервісного центру, підвищуючи ефективність їх роботи з прийому замовлень у центрі.

У спеціальному розділі було проаналізовано основні санітарно-гігієнічні вимоги до робочого місця користувача ПК.

Поставлені завдання виконано повністю, однак функціональність даної системи може бути збільшена за рахунок: інтеграція розробленої системи із основним сайтом сервісного центру, збільшення ролей, можливості підключення сканера штрих-кодів та прикріплення фото до замовлення.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Москаленко А. Інформаційно-довідкова система сервісного центру з ремонту обчислювальної техніки засобами C++/QT / А. Москаленко та ін.. Системи управління, навігації та зв'язку: збірник наукових праць. 2018. Т. 4, № 50. С. 129–134.

2. Крутова А. С. Моделювання інформаційної системи електронної торгівлі. Науковий вісник Ужгородського національного університету. Економіка. 2010. Вип. 29, ч. 1. С. 233–238.

3. Подход к оценке сложности диаграмм SADT (IDEF0) / А. А. Усков та ін. Международный журнал "Программные продукты и системы". 2015. Т. 27. С. 34–37.

4. Cheng-Leong A. Enactment of IDEF0 models. *International Journal of Production Research*. 1999. Vol. 37, no. 15. P. 3383–3397.

5. Модель Б. И. О распространении метода динамического программирования на класс неоднозначно детермированных бесконечношаговых процессов последовательного выбора решений: автореф. дис. канд. фіз.-мат. наук. Свердловск, 1978. 18 с.

6. Мова програмування VBA для інженерів: Навчальний посібник. Ivano-Frankivsk, Ukraine : ІФНТУНГ, 2019. 125 с.

7. Java / R. Toal et al. Programming Language Explorations. 2016. P. 119–144.

8. AB M. MySQL Developer's Guide. MySQL Press, 2007. 480 p.

9. Bassil Y. Autonomic HTML Interface Generator for Web Applications. *International journal of Web & Semantic Technology*. 2012. Vol. 3, №1. P. 33–47. URL: http://www.airccse.org/journal/ijwest/papers/3112ijwest04.pdf (дата звернення: 12.04.2022).

10. Apache Wicket free guide: A free user guide for Apache Wicket. Andrea Del Bene, 2013. 227 p.

11. What Is the ZK Ajax Framework?. *ZK™*. Berkeley, CA. P. 3–8. URL: https://link.springer.com/chapter/10.1007/978-1-4302-0440-4_1 (дата звернення: 13.02.2022).

12. Gutierrez F. Showing Your Spring Application on the Web. *Introducing Spring Framework*. Berkeley, CA, 2014. P. 133–144. URL: https://doi.org/10.1007/978-1-4302-6533-7_10 (дата звернення: 14.03.2022).

13. Linwood J., Minter D., Ottinger J. B. Beginning Hibernate: For Hibernate 5. Apress, 2016. 223 p.

14. Hunter J. Java servlet programming. 2nd ed. Beijing : O'Reilly, 2001. 753 p.

15. Loeliger J. Version control with Git. 2nd ed. Beijing : O'Reilly, 2012. 434 p.

16. Старух А., Депутат Б. Показники якості користувальницького програмного забезпечення. InterConf. 2021. С. 411–416.

17. Using EntityManager. *Beginning Database-Driven Application Development in Java™ EE*. Berkeley, CA, 2008. P. 253–282. URL: https://www.oreilly.com/library/view/java-ee-8/9781788833776/c08d22cd-2333-4671-b4a8-e1388b8e16ce.xhtml (дата звернення: 16.04.2022).

18. Санітарні норми мікроклімату виробничих приміщень : ДСН 3.3.6.042¬99-К. : МОЗ України, 2000. 42 с.

19. Жидецький В.Ц. Основи охорони праці: Підручник. – Львів: УАД, 2006. 336 с.

20. Менеджмент ризиків. Методи оцінки ризиків: вебсайт. URL: https://oppb.com.ua/news/oglyad-mizhnarodnyh-standartiv-z-upravlinnya-ryzykamy (дата звернення: 26.03.2022).

21. «Правила охорони праці під час експлуатації електронно-обчислювальних машин» 2010: вебсайт. URL: https://zakon.rada.gov.ua/laws/show/z0293-10 (дата звернення:25.05.2022)

22. Природне і штучне освітлення: ДБН В.2.5-28-2006., 2006.

ДОДАТОК А
Структурна схема бази даних