Міністерство освіти і науки України
Університет митної справи та фінансів

Факультет інноваційних технологій
Кафедра комп'ютерних наук та інженерії програмного забезпечення

Кваліфікаційна робота бакалавра

на тему «Створення модуля в системі "Журнал скарг" для фіксування скарг клієнтів, їх аналізу та покращення бізнес-процесів інтернет магазину»

Виконав: студент групи ІПЗ19-1
Спеціальність 121 Інженерія програмного забезпечення
Ванжа Микита Юрійович
(прізвище та ініціали)
Керівник к.т.н., доц. Ульяновська Ю.В
(науковий ступінь, вчене звання, прізвище та ініціали)
Рецензент Університет митної справи та фінансів
(місце роботи)
в.о. завідувача кафедри кібербезпеки та інформаційних технологій
(посада)
к.т.н., доц. Прокопович -Ткаченко Д.І.
(науковий ступінь, вчене звання, прізвище та ініціали)

Дніпро – 2023

АНОТАЦІЯ

Ванжа М.Ю. Створення модуля в системі "Журнал скарг" для фіксування скарг клієнтів, їх аналізу та покращення бізнес-процесів інтернет магазину.

Кваліфікаційна робота на здобуття освітнього ступеня бакалавр за спеціальністю 121 «Інженерія програмного забезпечення». – Університет митної справи та фінансів, Дніпро, 2023.

Зміст анотації

У результаті виконання дипломної роботи було проведено дослідження та розробку модуля "Журнал скарг" для інтернет-магазину. Основною метою було покращення бізнес-процесів та задоволення клієнтів шляхом зручної фіксації скарг, їх аналізу та автоматичного створення задач для вирішення.

Результати дослідження показали, що розробка модуля "Журнал скарг" має значний практичний потенціал. Він дозволяє клієнтам зручно відправляти скарги та спілкуватися з інтернет-магазином, що сприяє збереженню лояльності та задоволенню покупців. Крім того, модуль забезпечує фіксацію та аналіз даних про скарги, що дозволяє виявити проблемні місця в бізнес-процесах та покращити якість обслуговування.

Розроблений модуль "Журнал скарг" є важливим інструментом для покращення бізнес-процесів та задоволення клієнтів інтернет-магазину. Його впровадження сприятиме покращенню комунікації з клієнтами, виявленню проблемних ситуацій та вчасному прийняттю заходів для їх вирішення.

Ключові слова:

ООП, класи, ASP.Net, інтернет-магазин, клієнти, фіксація скарг, аналіз скарг, база даних, покращення якості обслуговування, комунікація з клієнтами, виявлення проблем, заходи для вирішення проблем, задоволення клієнтів, конкурентоспроможність.

ABSTRACT

Vanzha Mykyta. Creation of a module in the "Complaints Journal" system for for Capturing Customer Complaints, Analysis, and Improvement of Business Processes in an Online Store

Qualification work for obtaining a bachelor`s degree in specialty 121 “Software Engineering”. – University of Customs and Finance, Dnipro, 2023.

Abstract content:

This qualification work presents the research and development of a "Complaints Journal" module for an online store. The main objective was to enhance business processes and customer satisfaction through convenient capturing of complaints, their analysis, and automated task generation for resolution.

The research results demonstrate that the development of the "Complaints Journal" module holds significant practical potential. It enables customers to conveniently submit complaints and communicate with the online store, fostering loyalty and customer satisfaction. Additionally, the module facilitates the capture and analysis of complaint data, identifying areas for improvement in business processes and enhancing service quality.

The developed "Complaints Journal" module serves as a vital tool for enhancing business processes and customer satisfaction in the online store. Its implementation will improve communication with customers, identify problematic situations, and enable timely measures for resolution.

Keywords:

OOP, classes, ASP.Net, online store, customers, complaint capturing, complaint analysis, database, service quality improvement, customer communication, problem identification, problem resolution measures, customer satisfaction, competitiveness.

ЗМІСТ

ВСТУП

Практично всі інтернет магазини, які працюють зі споживачами, стикаються з скаргами та пропозиціями від своїх клієнтів. Найчастіше такі повідомлення надходять через телефонну лінію, електронну пошту або соціальні мережі. Це може призвести до затримок в обробці скарг та погіршення якості обслуговування клієнтів.

Актуальність роботи полягає в тому, що в сучасних умовах інтернет магазини стали дуже популярними серед населення, але при цьому часто виникають ситуації, коли клієнти не задоволені якістю наданих послуг або продуктів. Запити від клієнтів можуть надходити через різні канали комунікації, і важливо забезпечити ефективну систему обробки, щоб швидко відповідати на повідомлення та забезпечувати якісне обслуговування. У таких випадках важливо мати систему, яка зможе відслідковувати скарги клієнтів і вчасно реагувати на них.

Об'єктом роботи є вивчення методів фіксації скарг клієнтів в інтернет магазинах та їх подальшого опрацювання. Цей процес включає отримання повідомлень від клієнтів через різні канали комунікації, їх класифікацію та реєстрацію, подальшу обробку та вирішення проблем, пов'язаних із скаргами, а також аналіз отриманих пропозицій для поліпшення роботи магазину.

Предметом роботи є створення модуля в системі "Журнал скарг" для фіксування скарг клієнтів, їх аналізу та покращення бізнес-процесів інтернет магазину. Ця система повинна включати в себе засоби для ефективного отримання та класифікації повідомлень від клієнтів, зручний інтерфейс для роботи з цими повідомленнями, механізми для їх подальшої обробки та вирішення проблем, а також засоби для аналізу та використання пропозицій клієнтів для поліпшення роботи магазину.

Метою роботи є автоматизація процесу фіксування скарг клієнтів, їх аналізу та покращення бізнес-процесів інтернет-магазину шляхом створення модуля в системі "Журнал скарг". Основні завдання включають реалізацію

засобів для отримання та класифікації повідомлень, розробку зручного інтерфейсу для роботи з цими повідомленнями, розробку алгоритмів для подальшої обробки та вирішення проблем, а також інструменти для аналізу та використання пропозицій клієнтів. Метою є поліпшення процесу обробки скарг та пропозицій, зменшення часу реагування на клієнтські запити та підвищення задоволеності клієнтів від обслуговування в інтернет-магазині.

Для досягнення поставленої мети в дипломній роботі ставились та вирішувались наступні завдання:

- дослідити поточний процес фіксування скарг клієнтів, їх аналізу та покращення бізнес-процесів в інтернет-магазині. Це передбачало докладне вивчення існуючих процесів, виявлення недоліків та проблемних аспектів;
- проаналізувати потреби і вимоги клієнтів, а також виявити типові проблеми та тенденції, що виникають у зв'язку зі скаргами клієнтів. Це дозволило зрозуміти основні причини незадоволеності клієнтів та визначити напрямки для подальшого покращення;
- розробити модуль в системі "Журнал скарг", який забезпечує фіксування скарг клієнтів, їх аналіз та покращення бізнес-процесів інтернет-магазину.

Для виконання поставлених завдань були використані наступні методи:

- аналіз поточних процесів фіксування скарг клієнтів, що виявити недоліки та проблемні аспекти;
- проектування загальної форми для створення скарги з використанням технології Windows Forms;
- розробка функціональності модуля з використанням мови програмування C# та Windows Forms у платформі ASP.NET.Core;
- використання бібліотеки AutoMapper для перетворення об'єктів між різними типами та класами та бібліотеки FluentValidation для валідації даних;
- використання БД MySQL для зберігання даних та мови SQL для отримання даних.

Ці методи були використані для здійснення досліджень, збору та аналізу даних та розробки програмного забезпечення. Їх комбінація дозволила забезпечити комплексний підхід до вирішення проблеми фіксування скарг клієнтів, їх аналізу та покращення бізнес-процесів інтернет-магазину. Застосування такого комплексного підходу дозволить інтернет-магазину ефективно фіксувати скарги клієнтів, аналізувати їх та вносити відповідні покращення в бізнес-процеси

Основними завданнями проекту є: розробка модуля на мові програмування C# з використанням технології Windows Forms та ASP.NET. Розробка функціоналу для фіксування та збереження даних про скарги клієнтів, розробка механізму аналізу скарг та автоматичного створення задач для їх вирішення.

Отже, створення модуля для фіксування скарг клієнтів є ефективним інструментом для підтримки зв'язку з клієнтами та забезпечення високої якості обслуговування. Його використання сприяє покращенню репутації і конкурентоспроможності інтернет-магазину, а також забезпечує зростання задоволеності клієнтів.

Структура кваліфікаційної роботи: робота містить вступ, три розділи, висновки, 45 сторінок основного тексту, 19 рисунків. Перелік використаних джерел містить 15 посилань.

РОЗДІЛ 1 ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ. ПОСТАНОВКА ЗАВДАНЬ ДОСЛІДЖЕННЯ

1.1 Огляд методів фіксації та аналізу скарг клієнтів в інтернет магазинах

Один з найбільш ефективних методів фіксації скарг клієнтів - це використання онлайн форм для збору відгуків від клієнтів. Такі форми можуть містити поля для введення тексту, вибору категорій скарги, оцінки задоволення послугами тощо. Ці дані можуть бути зібрані в базу даних та використовуватись для аналізу та покращення бізнес-процесів.

Також можна використовувати соціальні мережі для фіксації скарг. Клієнти можуть залишати відгуки в коментарях під пости про товари або послуги. Для аналізу цих даних використовують спеціальні соціальні моніторингові інструменти, які дозволяють вимірювати ефективність рекламної кампанії, виявляти негативні відгуки та реагувати на них.

Один з методів фіксації скарг - це використання спеціальної системи управління відносинами з клієнтами Customer Relationship Management (CRM). Вона дозволяє фіксувати всі звернення клієнтів, включаючи скарги, і надавати відповідну інформацію співробітникам. Також можна проводити аналіз цих звернень та відстежувати їхній статус.

Система управління взаємовідносинами з клієнтами - є прикладним програмним забезпеченням для організацій, сприяє автоматизації стратегій взаємодії з клієнтами з метою підвищення рівня продажів та покращення обслуговування клієнтів. Головною метою впровадження CRM - є збільшення задоволеності клієнтів шляхом аналізу інформації про клієнтську поведінку, регулювання тарифної політики та використання інструментів маркетингу.

CRM базується на філософії, в якій центром бізнесу є клієнти, а головною метою діяльності компанії є забезпечення ефективного маркетингу, продажу та обслуговування клієнтів. Для підтримки цих бізнес-цілей збирається,

зберігається та аналізується інформація про споживачів, постачальників, партнерів та внутрішні процеси компанії. Функції підтримки бізнес-цілей включають продажі, маркетинг та підтримку споживачів.

CRM забезпечує ефективну та мінімально залучену до процесу роботи співробітників централізовану обробку даних, що дозволяє враховувати індивідуальні потреби замовників та виявляти ризики та можливості завдяки оперативній обробці інформації [1].

Ще одним методом аналізу скарг є метод NPS (Net Promoter Score), який дозволяє вимірювати задоволеність клієнтів та їхню готовність рекомендувати компанію іншим.

Його вимірюють, ставлячи «остаточне питання», яке дозволяє компаніям відслідковувати прихильників та недоброзичливців, здійснюючи чітку оцінку ефективності організації очима її клієнтів.

Вимірювання вашого Net Promoter Score дає вам число, яке ви можете регулярно збирати та відстежувати не тільки для всієї компанії, але й для кожного бізнесу, продукту, магазину чи команди обслуговування клієнтів.

Щоб розрахувати NPS, почніть із головного питання: «Наскільки ймовірно, що ви порекомендуєте нас другові чи колезі?» та оцініть відповіді за шкалою від нуля до десяти (рис. 1.1).

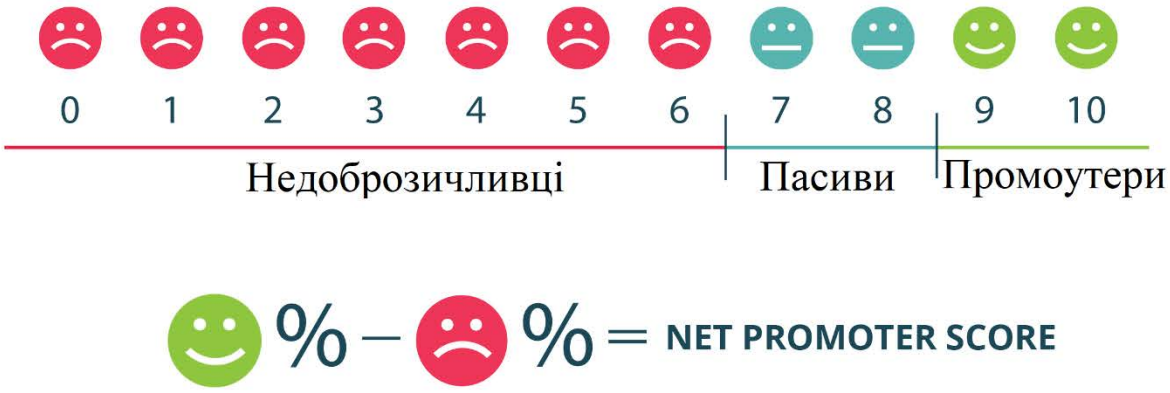

Рисунок 1.1 – розрахунок NPS

Ваш Net Promoter Score це просто відсоток клієнтів, які є прихильниками (ті, хто набрав 9 або 10 балів), мінус відсоток недоброзичливців (ті, хто набрав від 0 до 6).

За 11-бальною шкалою ми бачимо фактичні відмінності у поведінці тих, хто набирає дев'ятки та десятки, і тих, хто набирає нулі та шістки. Ці поведінкові відмінності пов'язані з економічною цінністю, справжньою силою розуміння вашого NPS.

Промоутери – віддані, захоплені фанати. Вони вихваляють компанію перед друзями та колегами. Вони з більшою ймовірністю, ніж інші, залишаться клієнтами і з часом збільшать свої покупки. Більше того, на них припадає понад 80% рефералів у більшості компаній. Загалом співробітникам приємно мати з ними справу.

Пасивні – ця група називається «пасивно задоволеною», тому що ця група задоволена поки що. Їхні ставки викупу та рефералів на 50% нижчі, ніж у промоутерів. Їхні реферали, швидше за все, будуть кваліфікованими та менш захопленими. Найпоказовіше: якщо реклама конкурента приверне їхню увагу, вони можуть відмовитись.

Недоброзичливці – незадоволені клієнти. На частку припадає понад 80% негативного сарафанного радіо. У них високі показники відтоку та дезертирства. Деякі з них можуть здатися прибутковими з точки зору бухгалтерського обліку, але їхня критика та погане ставлення підривають репутацію компанії, відлякують нових клієнтів та демотивують співробітників [2].

1.2 Аналіз існуючих систем журналів скарг

Наразі існує велика кількість CRM систем для аналізу та фіксування скарг клієнтів.

KeyCRM - українська CRM-система, що буде корисна будь-якому підприємцю (послуги, ecommerce, контакт-центри/онлайн-консультації, сфера краси і здоров'я), так як автоматизує процеси, що є майже у кожному бізнесі:

- комунікацію з клієнтами у месенджерах, соцмережах, по телефону, Email тощо;
- збір та обробку запитів (лідів), замовлень, записів на послугу з будь-яких джерел: сайти, сторінки Instagram, чати, інтернет-магазини та маркетплейси;
- роботу з воронками продажів, угод, проектів;
- аналітику та звітність щодо різних аспектів бізнесу, у тому числі по: фінансам, продажам, каналам (чатам), воронкам, ефективності менеджерів [3].

Система KeyCRM, як і будь-яка інша, має свої переваги та недоліки в контексті використання для журналу скарг.

Серед переваг можна виділити:

- розширені можливості аналітики та звітності, які допоможуть виявити та проаналізувати проблеми в роботі інтернет-магазину, що відображається в зростанні продажів та задоволеності клієнтів;
- відносно простий та зручний інтерфейс, що дозволяє швидко вивчати систему та з легкістю використовувати її;
- можливість інтеграції з іншими системами та сервісами, що значно полегшує роботу з журналом скарг та збільшує ефективність роботи в цілому.

Серед недоліків можна виділити:

- висока вартість системи;
- складність в налаштуванні та використанні, що може вимагати додаткових зусиль та ресурсів для роботи з системою;
- невеликий вибір готових шаблонів та рішень, що може обмежити можливості налаштування та персоналізації системи.

LiveAgent - це комплексна система для керування зв'язком з клієнтами, яка надає можливості для обробки звернень клієнтів з різних каналів комунікації,

включаючи електронну пошту, соціальні мережі, чат, телефон, а також для ведення журналу скарг і проблем клієнтів.

Сервіс дозволяє збирати та організовувати всі звернення клієнтів у єдиному інтерфейсі, що значно спрощує роботу з ними. Крім того, система надає широкий спектр інструментів для аналізу звернень клієнтів, статистики, рейтингів операторів, що дозволяє виявляти проблемні місця в роботі компанії та вдосконалювати бізнес-процеси [4].

Серед переваг LiveAgent можна виділити:

- єдиний інтерфейс для управління зверненнями з різних каналів комунікації;
- можливість швидко переходити від одного звернення до іншого;
- вбудовані інструменти для аналізу звернень та оцінки ефективності роботи операторів;
- Інтеграція з різноманітними сервісами.

Серед недоліків можна виділити:

- висока вартість планів підписки;
- складний процес налаштування та інтеграції з іншими сервісами;
- Складний інтерфейс для користувачів з меншим досвідом використання подібних систем;
- виникнення технічних проблем з підключенням до системи.

Creatio - єдина платформа для автоматизації бізнес-процесів для маркетингу, продажу та сервісу. Creatio (раніше відома як bpm'online) - це хмарна CRM-система, яка має широкий спектр функцій для автоматизації бізнес-процесів, включаючи обробку скарг клієнтів. Сервіс Creatio дозволяє вести журнал скарг, управляти запитами клієнтів, а також проводити аналіз ефективності комунікації з клієнтами [5].

До переваг системи Creatio можна віднести:

- інтеграцію з іншими бізнес-додатками. Creatio може бути легко інтегрована з іншими додатками, що дозволяє покращити ефективність роботи бізнесу;

- можливість автоматизувати процеси обробки скарг. Creatio дозволяє автоматизувати багато процесів, пов'язаних з обробкою скарг, що зменшує час на вирішення проблем клієнтів та збільшує їхню задоволеність;
- персоналізований підхід до клієнтів. Creatio надає можливість збирати та аналізувати інформацію про клієнтів, що дозволяє більш ефективно вирішувати проблеми та задовольняти їхні потреби.

Проте, до недоліків Creatio можна віднести:

- висока вартість – є досить дорогою системою, що може бути не доступною для деяких бізнесів.
- складність використання – має складний інтерфейс та вимагає певного рівня кваліфікації користувачів, що може бути проблемою для новачків.

1.3 Покращення бізнес-процесу обслуговування клієнтів в інтернет магазинах

Бізнес-процес являє собою систему послідовних, цілеспрямованих і регламентованих видів діяльності, у якій за допомогою керуючого впливу і необхідних ресурсів входи процес перетворюються у виходи, результати процесу, що представляють цінність для споживачів [6].

Інтернет-магазини мають багато різних бізнес-процесів, що їх потрібно оптимізувати для забезпечення ефективної роботи магазину та задоволення клієнтів. Основні бізнес-процеси в інтернет-магазинах включають такі етапи:

1. Обробка замовлень: включає прийом замовлення, обробку оплати, підготовку товарів до відвантаження та їх відправлення;
2. Інвентаризація товарів: це процес контролю за наявністю товарів на складі, визначення потреб у додаткових поставках та планування замовлень;

3. Обслуговування клієнтів: це процес взаємодії з клієнтами, який включає прийом замовлень, відповіді на запитання та скарги, обробку повернень та зворотній зв'язок з клієнтами;

4. Маркетинг та продажі: цей процес включає рекламу товарів, вивчення потреб клієнтів та збільшення продажів;

5. Аналітика та відстеження: цей процес включає збір даних про клієнтів та їхній покупковий профіль, відстеження ефективності рекламних кампаній та процесів продажу [7].

Існує кілька способів покращення бізнес-процесу обслуговування клієнтів та рішення скарг в інтернет магазинах:

1. Швидкий та ефективний збір та обробка інформації про скарги клієнтів. Для цього можна використовувати спеціальні програми або CRM-системи, які дозволяють автоматизувати процес збору та аналізу скарг. Це дозволяє швидко реагувати на проблеми та забезпечувати якісне обслуговування клієнтів;

2. Розвиток системи повернення товару. Це дозволить знизити кількість скарг та покращити відносини з клієнтами. Для цього потрібно розробити простий та зрозумілий процес повернення товару та надати клієнтам можливість вільно повертати товари, які не відповідають їх очікуванням;

3. Вдосконалення комунікації з клієнтами. Важливо забезпечити якісну комунікацію з клієнтами, щоб вони могли отримувати вчасну та точну інформацію про стан свого замовлення або повернення товару. Для цього можна використовувати різні канали комунікації, такі як телефон, електронна пошта, соціальні мережі тощо;

4. Покращення якості обслуговування. Важливо забезпечити якісне обслуговування клієнтів, яке відповідає їх очікуванням та потребам. Для цього необхідно забезпечити високу кваліфікацію персоналу та відповідність їх роботи стандартам якості обслуговування;

5. Використання аналітики та зворотного зв'язку. Для покращення бізнес-процесу обслуговування клієнтів з рішенням скарг, важливо використовувати аналітику та зворотний зв'язок.

1.4 Висновки до першого розділу. Постановка завдання досліджень

В цьому розділі було проведено детальний аналіз методів фіксації та аналізу скарг клієнтів в інтернет-магазинах, а також огляд існуючих систем журналів скарг. Огляд існуючих систем журналів скарг показав, що багато з них мають обмежені можливості та не враховують специфіку інтернет-магазинів.

Актуальність роботи полягає в тому, що в сучасних умовах інтернет магазини стали дуже популярними серед населення, але при цьому часто виникають ситуації, коли клієнти не задоволені якістю наданих послуг або продуктів. У таких випадках важливо мати систему, яка зможе відслідковувати скарги клієнтів і вчасно реагувати на них.

В ході даної роботи необхідно розробити «модуль скарг» для системи засновану на моделі управління взаємовідносинами з клієнтами, так як було зроблено висновок про те, що більшість існуючих систем журналів скарг надають зручні інструменти для фіксації та відстеження скарг клієнтів, але не всі дозволяють проводити аналіз та покращення бізнес-процесів, пов'язаних з обробкою цих скарг та не можливостю інтегрувати в систему.

Метою роботи є створення ефективної системи обробки скарг та пропозицій в інтернет-магазинах, яка забезпечуватиме швидке реагування на повідомлення клієнтів та поліпшення обслуговування.

Для досягнення поставленої мети в дипломній роботі ставились та вирішувались наступні завдання:

- дослідити поточний процес фіксування скарг клієнтів, їх аналізу та покращення бізнес-процесів в інтернет-магазині. Це передбачало докладне вивчення існуючих процесів, виявлення недоліків та проблемних аспектів;
- проаналізувати потреби і вимоги клієнтів, а також виявити типові проблеми та тенденції, що виникають у зв'язку зі скаргами клієнтів. Це дозволило зрозуміти основні причини незадоволеності клієнтів та визначити напрямки для подальшого покращення;
- розробити модуль в системі "Журнал скарг", який забезпечує фіксування скарг клієнтів, їх аналіз та покращення бізнес-процесів інтернет-магазину.

Проблема:

На даний момент немає інструменту для фіксації скарг клієнтів про неякісну доставку або роботу співробітників. Відсутність такого інструменту не дає можливості аналізувати та контролювати скарги клієнтів, а також через відсутність класифікації та статистичної інформації щодо скарг немає можливості приймати рішення щодо покращення бізнес-процесів.

Рішення:

Створити в системі журнал скарг, де співробітники зможуть фіксувати скарги клієнтів та вказувати рішення щодо них.

Ціль:

Налагодити облік та забезпечити можливість контролю за врегулюванням претензій клієнтів.

Розробка модуля дозволить поліпшити процес обробки скарг та пропозицій в інтернет-магазинах, забезпечить швидке реагування на повідомлення клієнтів та підвищити задоволеність клієнтів від обслуговування.

РОЗДІЛ 2 АНАЛІЗ ПРОГРАМНИХ ЗАСОБІВ ДЛЯ РЕАЛІЗАЦІЇ МОДУЛЯ СКАРГ

2.1 Вибір програмних засобів для реалізації модуля

При розробці модуля скарг необхідно обрати програмне забезпечення, яке забезпечить ефективну реалізацію. Для розробки модуля скарг була обрана технологія ASP.NET Core.

ASP.NET Core – є кросплатформовим, високопродуктивним середовищем з відкритим вихідним кодом для створення сучасних хмарних програм, підключених до Інтернету [8]. Вона також пропонує зручні інструменти для розробки веб-сервісів і API.

Однією з ключових переваг ASP.NET Core є його висока продуктивність. Це досягається завдяки оптимізації та використанню сучасних технологій. ASP.NET Core здатний ефективно обробляти великі обсяги запитів та забезпечувати швидку віддачу даних користувачам.

Важливою особливістю ASP.NET Core є його масштабованість. Він надає механізми автоматичного масштабування програм залежно від навантаження. Це дозволяє модулю скарг ефективно працювати як з невеликою кількістю користувачів, так і з високими навантаженнями при інтенсивному використанні системи [9].

Для розробки контролерів та обробки HTTP-запитів у модулі використовується бібліотека Microsoft.AspNetCore.Mvc.

ASP.NET MVC є багатофункціональною платформою для створення веб-додатків та API-інтерфейсів за допомогою структури проектування Model-View-Controller [10]. Вона надає необхідні класи та інструменти для реалізації контролерів, дій та маршрутизації запитів. Завдяки цій бібліотеці розробники можуть ефективно обробляти запити та створювати веб-інтерфейс для взаємодії зі скаргами.

Для забезпечення авторизації та керування доступом до ресурсів програми можна використовувати простір імен Microsoft.AspNetCore.Authorization. Авторизація ASP.NET Core надає просту декларативну роль та багатофункціональну модель на основі політик [11]. Він дозволяє визначати політики авторизації та контролювати доступ до певних функцій та даних залежно від прав користувачів. Це забезпечує безпеку та захист конфіденційних даних.

2.2 Зберігання даних в системі та засоби роботи з ними

Зберігання даних є необхідною складовою будь-якої програмної системи, в тому числі й модуля скарг. Основні причини, чому потрібно зберігати дані в системі та використовувати засоби роботи з ними, такі:

- Збереження інформації: дані, отримані від користувачів, мають бути збережені для подальшого використання. Зберігання даних дозволяє зберегти інформацію на тривалий час і зробити її доступною для обробки, аналізу, звітності та інших операцій.
- Обробка та аналіз даних: збережені дані можуть бути використані для різноманітних операцій, таких як пошук, фільтрація, сортування, обчислення агрегованих значень, створення звітів, аналітики тощо. Це дозволяє виконувати складні завдання з обробки і аналізу даних, що забезпечує підтримку функцій модуля скарг, таких як пошук скарг за певними критеріями або генерація звітів.
- Забезпечення доступу до даних: збереження даних в системі дозволяє забезпечити доступ до них користувачам, які мають відповідні права доступу. Це дозволяє здійснювати роботу з даними, редагувати їх, отримувати інформацію, а також контролювати доступ до конфіденційної інформації.

Загалом, зберігання даних та засоби роботи з ними є необхідними для забезпечення функціональності модуля скарг. Вони дозволяють зберігати,

обробляти та аналізувати – це сприяє ефективній роботі з скаргами та надає можливість користувачам отримувати відповідну інформацію та роз'яснення.

Використання бази даних MySQL є одним з можливих варіантів для зберігання та керування скаргами.

MySQL - це система керування базами даних. У реляційній базі даних дані зберігаються не всі скопом, а в окремих таблицях, завдяки чому досягається виграш в швидкості і гнучкості. Таблиці зв'язуються між собою за допомогою відносин, завдяки чому забезпечується можливість об'єднувати при виконанні запиту дані з декількох таблиць. SQL як частина системи MySQL можна охарактеризувати як мову структурованих запитів плюс найбільш поширений стандартний мова, яка використовується для доступу до баз даних [12].

Основні ролі та значення бази даних MySQL у реалізації модуля скарг включають наступне:

- зберігання скарг: база даних MySQL використовується для зберігання всіх даних, пов'язаних з скаргами. Це включає інформацію про саму скаргу (текст, дата подання, автор і т. д.), а також додаткові деталі, які можуть бути пов'язані зі скаргою (наприклад, тип скарги, джерело, статус тощо);
- організація даних: база даних MySQL надає структурований підхід до організації даних. Таблиці бази даних можуть бути створені для різних сутностей модуля скарг, таких як скарги, типи скарг, джерела скарг, статуси скарг тощо. Це дозволяє зберігати дані в логічно впорядкованій формі та спрощує їхнє управління;
- керування доступом: MySQL надає механізми керування доступом до даних. Це дозволяє встановлювати права доступу до таблиць і полів бази даних для різних користувачів модуля скарг. Наприклад, адміністратор може мати повний доступ до всіх даних, тоді як звичайний користувач може мати обмежений доступ лише для перегляду і деяких змін.
- запити і звіти: база даних MySQL надає мову запитів SQL, яка дозволяє виконувати різні запити до даних. Це включає виділення даних за

певними умовами, сортування, фільтрацію тощо. Це дає можливість реалізувати різноманітні функції модуля скарг, наприклад, пошук і виведення скарг за певними критеріями, створення звітів або аналітики щодо скарг;

- масштабованість: база даних MySQL дозволяє масштабувати роботу модуля скарг. Вона може обробляти велику кількість даних та запитів, що робить її придатною для проектів різного розміру і обсягу;
- резервне копіювання та відновлення даних: MySQL надає можливість резервного копіювання бази даних, що є важливим для забезпечення безпеки і можливості відновлення даних в разі виникнення непередбачуваних ситуацій, таких як втрата даних або системний збій.

Щоб отримати дані з бази даних, використовується мова SQL (Structured Query Language – мова структурованих запитів). SQL – це мова програмування для роботи з наборами фактів і зв'язками між ними [13].

Основні методи для роботи з базою даних MySQL у контексті модуля скарг можуть включати:

- вставка даних за допомогою SQL-запитів або програмного інтерфейсу до бази даних можна вставляти нові записи про скарги, типи скарг, джерела скарг і статуси скарг;
- оновлення даних за допомогою SQL-запитів можна оновлювати існуючі записи з інформацією про скарги, типи скарг, джерела скарг і статуси скарг;
- вибірку даних за допомогою SQL-запитів можна вибирати дані з таблиць бази даних. Наприклад, можна отримати список усіх скарг, відфільтрувати їх за певними критеріями (наприклад, за типом, статусом або датою подання) і отримати відповідні результати;
- зв'язки між таблицями: зазначені таблиці мають зв'язки між собою за допомогою зовнішніх ключів. Це дозволяє здійснювати зв'язані операції, наприклад, отримувати інформацію про тип скарги або джерело скарги, пов'язані з конкретною скаргою;

- запити і фільтрація за допомогою SQL-запитів можна виконувати складні операції, які включають об'єднання таблиць, сортування, групування і обчислення агрегатних функцій.

2.3 Засоби для валідації даних

Засоби валідації даних відіграють важливу роль у розробці програмного забезпечення. Вони надають механізми для перевірки вхідних даних на відповідність певним правилам та обмеженням.

Валідація даних дозволяє гарантувати, що вхідні дані відповідають очікуваним форматам та вимогам. Це допомагає запобігти введенню некоректних або неприпустимих даних, які можуть призвести до помилок, неправильних результатів або порушення цілісності системи.

Валідація даних допомагає надати кращий користувальницький досвід, попереджаючи користувачів про некоректні дані або неправильно заповнені поля. Повідомлення про помилки та підказки, що надаються в процесі валідації, допомагають користувачам виправити помилки та вводити дані правильно з першого разу.

Засоби валідації дозволяють контролювати та обмежувати дані, які можуть бути введені або оброблені у системі. Наприклад, можна визначити обмеження на довжину рядка, допустимі символи, числові діапазони або формати дати. Це допомагає підтримувати правильність даних та запобігати помилкам через некоректні дані.

У результаті засоби валідації даних є важливою частиною розробки програмного забезпечення, забезпечуючи цілісність. даних, забезпечуючи безпеку, покращуючи користувальницький досвід, контролюючи дані та підвищуючи ефективність системи.

Без засобів валідації даних програма може мати проблеми, такі як некоректні дані, невідповідність форматів, неправильні результати або помилки в обробці. Користувачі можуть зіткнутися з труднощами під час заповнення форм, отримувати незрозумілі повідомлення про помилки або стикатися з некоректними результатами.

При розробці модуля скарг була обрана бібліотека FluentValidation для обробки валідації даних. FluentValidation – це популярна бібліотека .NET для створення строго типізованих правил перевірки. При створенні програми, яка надає API для інших систем або зовнішній інтерфейс для користувачів, ви хочете переконатися, що дані, введені або проаналізовані у вашому додатку, дійсні.

Використовуючи бібліотеку .NET, таку як FluentValidation, ви можете легко налаштувати правила перевірки за допомогою лямбда-виразів. Бібліотека навіть дозволяє вам ввести клієнту повідомлення про помилку з подробицями про те, чому перевірка не вдалася [14].

Вона має ряд переваг, таких як:

- пропонує більш читаний та гнучкий підхід до визначення правил валідації;
- дозволяє розробникам чітко визначити правила перевірки даних та забезпечує зручний спосіб інтеграції з моделями уявлень у ASP.NET Core.

Це допомагає створити надійний та безпечний модуль для обробки скарг, де дані проходять необхідну валідацію перед обробкою та збереженням. Використання бібліотеки FluentValidation у модулі скарг дозволяє легко визначити правила валідації для різних полів та компонентів, таких як тексти, дати, числа та інші дані, що вводяться.

2.4 Засоби для перетворення об'єктів між різними типами та класами

Засоби перетворення об'єктів між різними типами і класами є важливою складовою розробки додатків. Вони надають зручні інструменти для перетворення даних та забезпечують гнучкість та ефективність при роботі з різними об'єктами.

Під час розробки журналу скарг було використано бібліотеку AutoMapper, щоб забезпечити зручне перетворення об'єктів між різними типами та класами.

AutoMapper — це проста маленька бібліотека, створена для вирішення оманливо складної проблеми — позбавлення від коду, який зіставляв один об’єкт з іншим [15]. Він дозволяє автоматично копіювати значення з одного об'єкта в інший, ґрунтуючись на певних правилах чи конфігурації.

Переваги використання AutoMapper у журналі скарг включають:

1. Спрощення коду: AutoMapper дозволяє уникнути необхідності вручну прописувати безліч перетворень між об'єктами. Це скорочує обсяг коду та спрощує його читання та розуміння;

2. Скорочення часу розробки: Завдяки автоматичному зіставленню полів та властивостей AutoMapper спрощує та прискорює процес перетворення даних між об'єктами різних типів. Розробнику не потрібно писати безліч коду, що повторюється, для виконання цих перетворень;

3. Гнучкість та настроюваність: AutoMapper надає можливість налаштовувати правила зіставлення, щоб врахувати особливості кожного конкретного випадку. Це включає ігнорування певних полів, налаштування перетворення значень та інші аспекти перетворення;

4. Підтримка складних перетворень: AutoMapper має потужні можливості для обробки складних сценаріїв перетворення даних. Це включає зіставлення колекцій, вкладених об'єктів, перетворення імен полів та інші додаткові функції;

5. Широка підтримка та спільнота: AutoMapper є популярною бібліотекою у спільноті розробників на платформі .NET. Його активна спільнота забезпечує хорошу підтримку, оновлення та наявність документації, а також доступ до різних ресурсів та прикладів використання.

Використання AutoMapper у журналі скарг допомагає спростити та прискорити процес перетворення даних між різними типами та класами. Це підвищує ефективність розробки та покращує підтримуваність коду.

2.5 Висновки до другого розділу.

У процесі розробки модуля скарг було обрано такі програмні інструменти та технології:

ASP.NET Core: Ця технологія була обрана як основна платформа для розробки модуля скарг. ASP.NET Core пропонує безліч переваг, включаючи високу продуктивність, масштабованість, крос-платформність та підтримку асинхронної обробки запитів.

Microsoft.AspNetCore.Mvc: Бібліотека Microsoft.AspNetCore.Mvc була використана для розробки контролерів та обробки HTTP запитів у модулі. Вона надає необхідні класи та інструменти для реалізації контролерів, дій та маршрутизації запитів, що дозволяє ефективно обробляти запити та створювати веб-інтерфейс для взаємодії зі скаргами.

FluentValidation: Бібліотека FluentValidation була обрана для забезпечення валідації даних. Вона надає зручні та гнучкі засоби для перевірки вхідних даних на відповідність заданим правилам та обмеженням. Її використання дозволяє забезпечити коректність та цілісність даних, що є важливим аспектом розробки програмного забезпечення.

MySQL: база даних MySQL є підходящим інструментом для зберігання та керування даними модуля скарг. Вона дозволяє ефективно організувати дані, забезпечує надійність, безпеку та масштабованість, що є важливими аспектами розробки такого модуля.

AutoMapper: AutoMapper був використаний як засіб для перетворення об'єктів між різними типами та класами. Він надає зручні методи для автоматичного зіставлення властивостей об'єктів, що дозволяє спростити та прискорити процес перетворення даних. Використання AutoMapper сприяє покращенню продуктивності та читання коду при роботі з різними об'єктами.

Вибір даних програмних засобів та бібліотек обумовлений їх перевагами та можливостями, які вони надають для реалізації модуля скарг. ASP.NET Core забезпечує ефективність, масштабованість та крос-платформність для створення сучасних веб-додатків. Бібліотека Microsoft.AspNetCore.Mvc надає зручні інструменти для розробки контролерів, дій та маршрутизації запитів. FluentValidation забезпечує гнучку та централізовану валідацію даних. AutoMapper дозволяє спростити перетворення об'єктів між різними типами та класами.

Водночас ці засоби та бібліотеки утворюють сильний стек інструментів для реалізації модуля скарг, забезпечуючи ефективність, надійність та гнучкість при розробці та обробці даних.

РОЗДІЛ 3 РОЗРОБКА МОДУЛЯ СКАРГ ДЛЯ ДОДАТКУ TELEMART.CLIENT

3.1 Технічне завдання

Модуль "Журнал скарг"

Розташувати в блоці магазин останнім пунктом, доступ є у всіх.

Загальна форма

На панелі інструментів мають відображатися такі інструменти:

1. Оновити
2. Колонки
3. Додати
4. Змінити
5. Довідка

Таблична частина повинна містити такі поля:

- Номер
- Задача
- Замовлення
- Заявка
- Створено
- Дедлайн
- Пріоритет
- Джерело
- Відповідальний
- Контрагент
- Тип
- Статус (нова, врегульована, відхиллено)

Параметри пошуку мають бути такими:

- Створено
- Скарга (номер скарги)
- Замовлення (номер замовлення)
- Серв. заявка (номер заявки)
- Тип (що випадає дерево як у дзвінках)
- Відповідальний
- ПІБ
- Телефон
- Статус (за замовчуванням лише Нова)

Форма "Створення скарги"

Спочатку користувач вибирає причину скарги із 3 пунктів, а саме:

- По замовленню;
- По сервісній заявці
- Інше

Потім з'являється поле для введення інформації:

- Джерело (Заява\E-mail\Дзвінок\Відгук на сайті\Відгук на іншому ресурсі\Чат)
- Тип (Товар \Рівень обслуговування\ Технічні складності\ Контактність інтернет-магазину\Оплати замовлення\ Доставка\ Сервісне обслуговування\ Інше)
- Пріоритет
- Скарга
- Відповідальний
- Дедлайн
- ПІБ
- Телефон
- Електронна пошта

Форма "Перегляд скарги"

Основні поля:

- Замовлення - по кліку на лупу відкривати форму перегляду замовлення.
- Товар – по кліку на лупу відкривати форму перегляду товару, якщо поле заповнено.
- Задача в B24
- Пріоритет
- Відповідальний
- ПІБ
- Телефон 1 та 2
- Текст скарги

Особливі вимоги:

1. Усього може бути 3 статуси:

- Нова
- Врегульована
- Відхиллено

2. Система має створювати нові скарги у статусі "Нова".

3. Система повинна дати користувачеві можливість змінити статус скарги на "Врегульована" або "Відхилена" лише для скарг із статусом "Нова".

4. При натисканні на кнопку "Врегульована"/"Скасована" система повинна додатково запитати користувача підтвердження зміни статусу.

5. На етапі створення скарги система повинна за замовчуванням заповнювати реквізити, згідно з обраною заявкою, з можливістю редагування користувачем.

3.2 Розробка форми модуля

Було розроблено форму «Журнал скарг» (рис 3.1) зі всіма полями які були прописані в ТЗ, ці поля можна скривати за необхідністю через «Колонки» (рис 3.2)

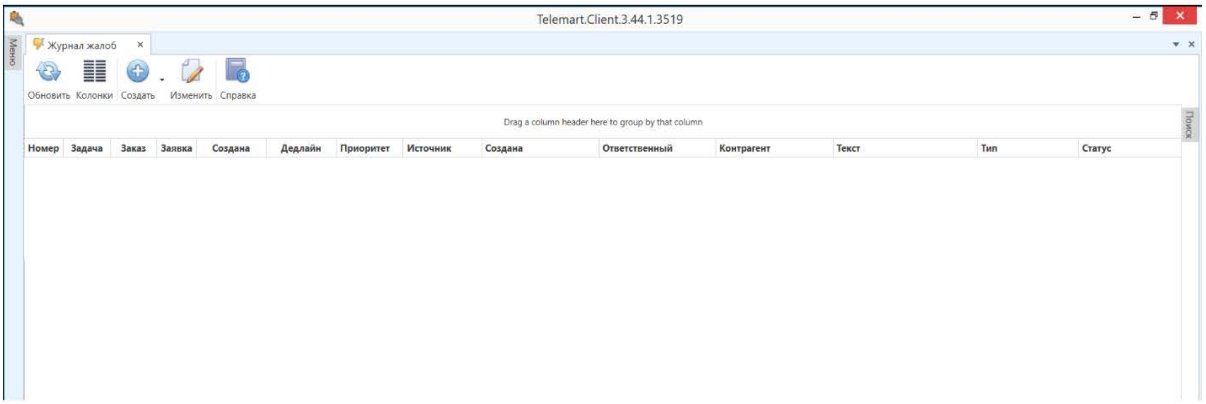

Рисунок 3.1 – модуль «Журнал скарг»

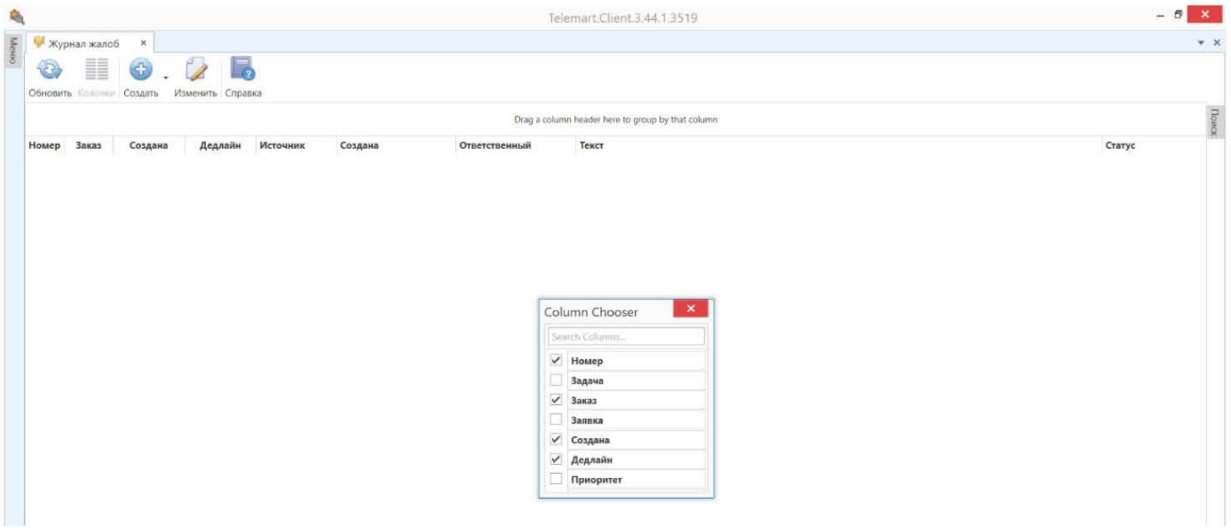

Рисунок 3.2 – приховання полів форми

При натисканні кнопки створити, з'являється випадаючий список, в якому необхідно обрати звідки прийшла скарга, це може бути по замовлення, по сервісній заявці або інше (рис 3.3).

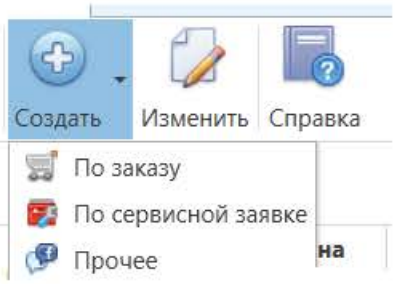

Рисунок 3.3 – випадаючий список кнопки «Створити»

Обравши пункт «Інше», з'являється форма для створення скарги (рис 3.4), де необхідно заповнити поля.

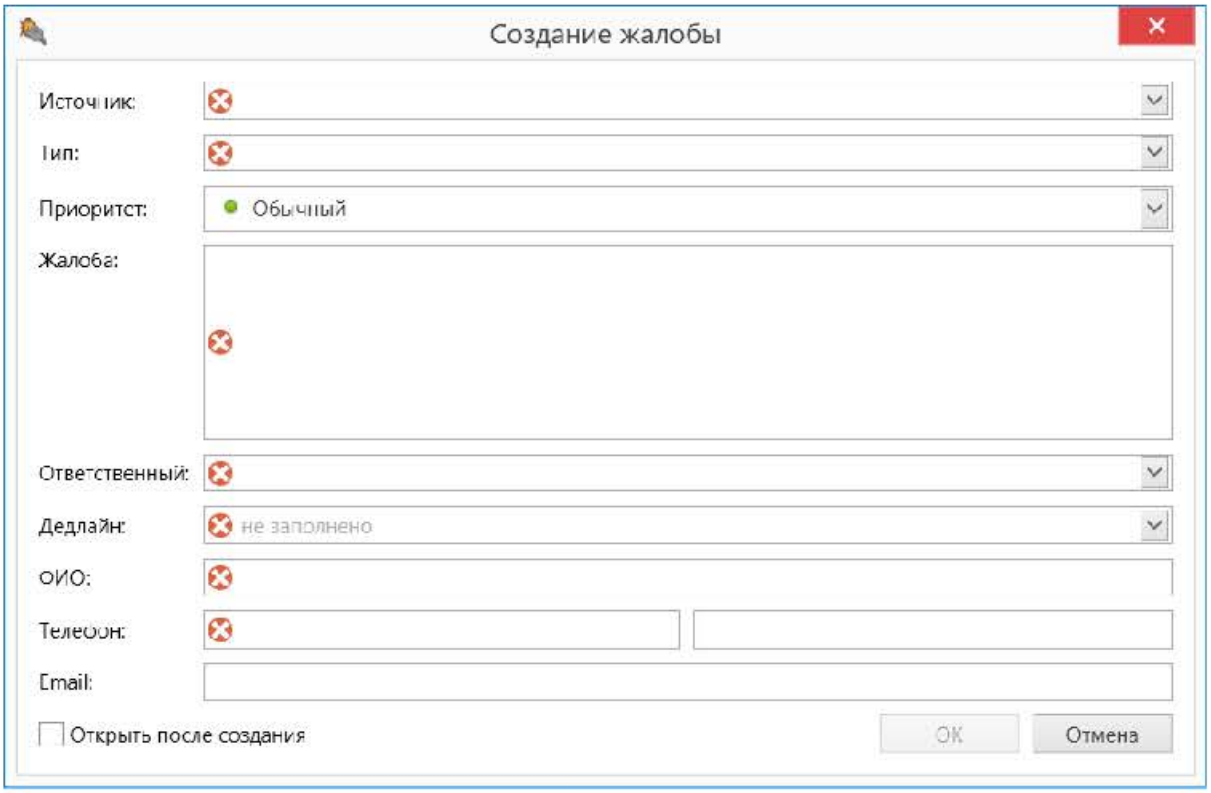

Рисунок. 3.4 – форма «Створення скарги»

Вікно пошуку скарги (рис 3.5) – в цьому вікні можна знайти скаргу по заказу, заявці, по типу, за номером телефону та тощо.

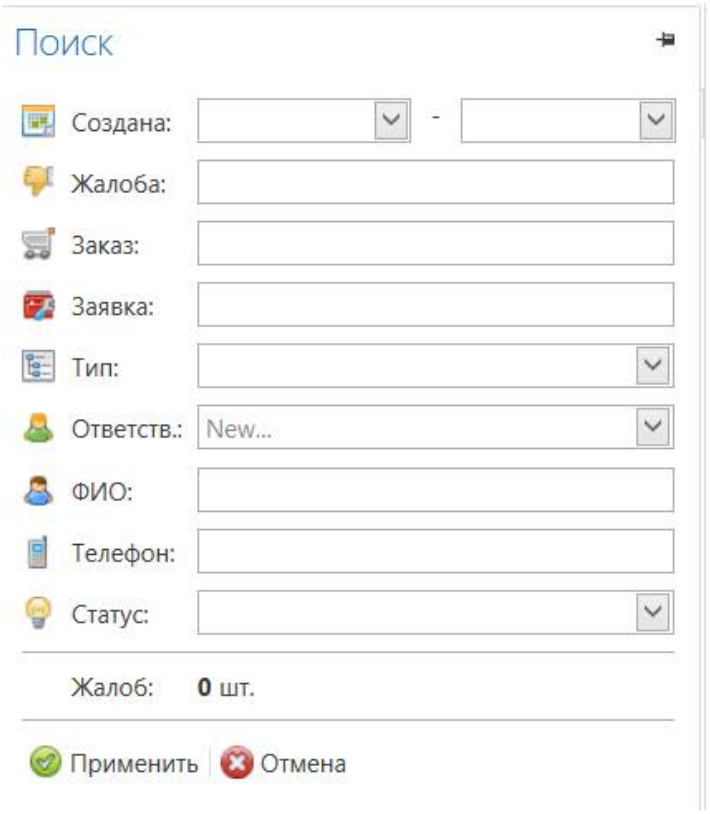

Рисунок 3.5 – вікно пошуку

Форма перегляд скарги (рис 3.6) – в цьому вікні можна редагувати скаргу.

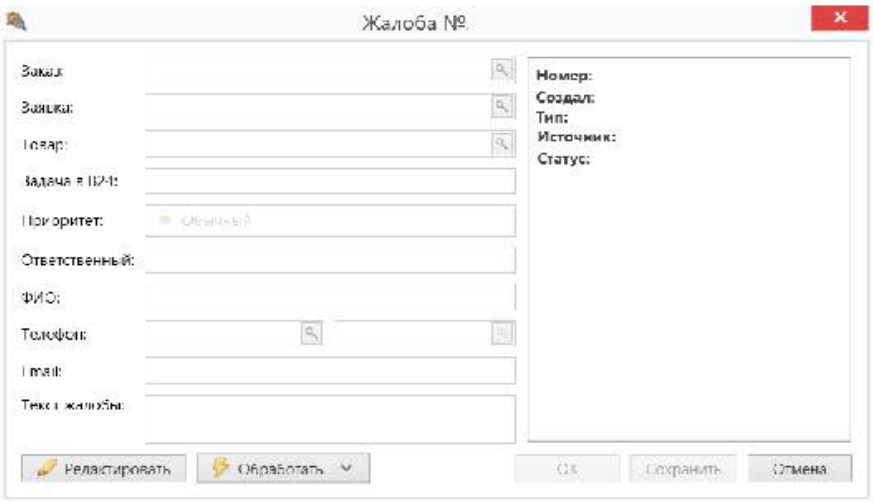

Рис 3.6 – форма «Перегляд скарги»

3.3 Серверна частина

Для зберігання скарг були створені окремі таблиці в базі даних. Таблиця "tm_complaint" (рис 3.7) зберігає основну інформацію про скарги, включаючи їхній унікальний ідентифікатор, ідентифікатор користувача, текст скарги, дату подання та посилання на тип скарги, джерело скарги, статус скарги тощо. Використання зовнішніх ключів у таблиці дозволяє пов'язувати дані з таблиць "tm_complaint_type" (рис 3.8), "tm_complaint_source" (рис 3.9) і "tm_complaint_state" (рис 3.10) для отримання додаткової інформації про тип, джерело і статус скарги.

Рисунок 3.7 – структура таблиці tm_complaint

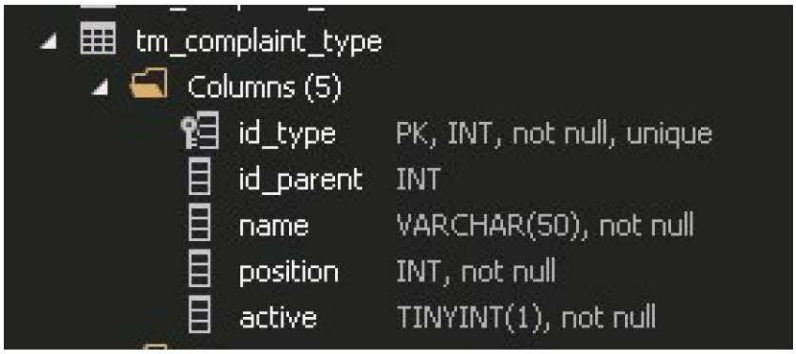

Рисунок 3.8 – структура таблиці tm_complaint_type

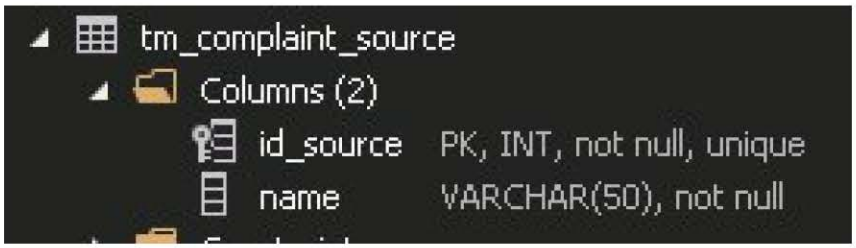

Рисунок 3.9 – структура таблиці tm_complaint_source

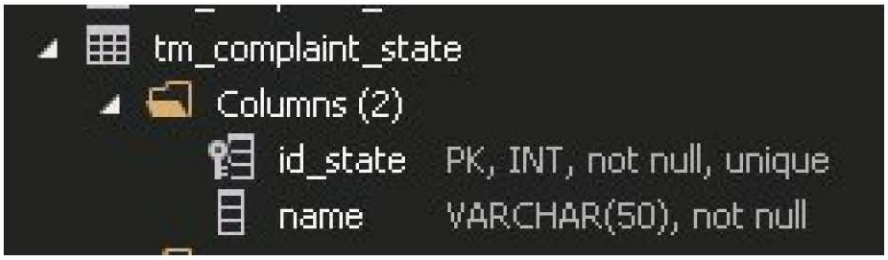

Рисунок 3.10 – структура таблиці tm_complaint_state

Така структура бази даних дозволяє ефективно зберігати і керувати інформацію. Використання бази даних MySQL для зберігання цих даних дозволяє ефективно організувати, зберігати та керувати інформацією про скарги.

Для реалізації модуля скарг було використано три класи, які відіграють ключову роль у його функціонуванні:

1. ClientViewModel – цей клас є модель даних для подання інформації про скаргу. Він містить такі властивості, як ідентифікатор скарги, дата подання, опис, статус та інші атрибути, необхідні для зберігання та обробки даних про скаргу. Класи моделі представлення клієнта представляють дані, які передаються

між контролерами та уявленнями (шаблонами) у додатку. Вони служать для передачі даних між інтерфейсом користувача і бізнес-логікою програми.

2. Controller – цей клас представляє контролер, відповідальний за обробку запитів HTTP, пов'язаних зі скаргами. Контролер визначає відповідь, яку користувач надсилає, коли користувач надсилає запит браузера [16]. Він містить методи для створення нової скарги, отримання інформації про існуючу скаргу, оновлення статусу скарги та інших операцій, пов'язаних із керуванням скаргами. Усередині цих методів використовуються екземпляри класів сервісів для виконання відповідних операцій із даними.

3. Service – цей клас представляє сервіс, відповідальний за бізнес-логіку, пов'язану зі скаргами. Він містить методи для взаємодії з базою даних, виконання операцій CRUD (створення, читання, оновлення, видалення) для об'єктів Complaint, а також інші методи обробки та перевірки даних, валідації та перетворення об'єктів.

Взаємодія між цими класами забезпечує функціональність модуля скарг. Клас ClientViewModel використовується для подання даних про скарги, клас Controller обробляє HTTP-запити та делегує операції сервісу Service, який у свою чергу виконує операції з базою даних та бізнес-логіку. Така архітектура дозволяє розділити окремі відповідальності та спростити розробку, тестування та підтримку модуля скарг.

Основні методи та поля класів ClientViewModel та Service – вони будуть ідентичні, окрім методу CreateAsync. У класі ClientViewModel додатково створюється завдання в CRM системі Bitrix24 (рис 3.7).

```
if (employee?.BitrixId != null && WorkContext.Employee.BitrixId != null)
{
    OperationResult<int?> result = await BitrixTaskHelper.CreateAsync(
        $"Решение жалобы № {complaint.Id}",
        BuildTaskBody(complaint),
        complaint.Deadline,
        WorkContext.Employee.BitrixId.Value,
        employee.BitrixId.Value,
        BitrixEmployeeOptions.BitrixComplaintWorkGroupId);

    if (result.IsSuccess)
    {
        complaint.BitrixId = result.Data;
        await UnitOfWork.SaveChangesAsync();
    }
    else
    {
        warning = $"Не удалось создать задачу в Битрикс24. {result.Error}";
    }
}
else
{
    warning = "Не удалось создать задачу в Битрикс24. У текущего пользователя или ответственного нет аккаунта в Битрикс24.";
}
```

Рисунок 3.7 – створення завдання в Bitrix24 у класі ClientViewModel

- Конструктор public ComplaintService(...) - конструктор класу ComplaintService, який приймає кілька залежностей (інтерфейсів) та ініціалізує відповідні приватні поля. Він також викликає конструктор базового класу ServiceLockableBase<Complaint, ComplaintDto>, передаючи йому деякі залежності. Конструктор є точкою входу під час створення об'єкта ComplaintService.
- private IWorkContext WorkContext – приватне поле, що представляє інтерфейс IWorkContext, який надає інформацію про поточного користувача або робочий контекст.
- private IComplaintTypeRepository ComplaintTypeRepository – приватне поле, що представляє інтерфейс IComplaintTypeRepository, який надає доступ до сховища типів скарг.
- private IBitrixTaskHelper BitrixTaskHelper – приватне поле, що представляє інтерфейс IBitrixTaskHelper, який надає методи роботи з завданнями в Бітрікс24.
- private BitrixEmployeeOptions BitrixEmployeeOptions – приватне поле, що представляє налаштування для роботи зі співробітниками в Бітрікс24.
- Public sealed class ComplaintService: ServiceLockableBase <Complaint,ComplaintDto>, IComplaintService – визначення класу

ComplaintService, який успадковується від базового класу ServiceLockableBase і реалізує інтерфейс IComplaintService.

- public override int GetCurrentEmployeeId()– перевизначення методу GetCurrentEmployeeId() з базового класу. Метод повертає ідентифікатор співробітника.
- public async Task<Result> CreateAsync (ComplaintCreateDto dto) – метод для створення нової скарги. Приймає об'єкт ComplaintCreateDto, який містить дані для створення скарги. Метод виконує деякі перевірки, створює новий об'єкт скарги (Complaint), зберігає їх у базі даних, і повертає результат операції як об'єкта Result.
- public async Task<ComplaintDto> GetOneAsync(int id) – метод який асинхронно повертає одну скаргу за вказаним ідентифікатором (id). Всередині методу викликається метод GetEntityAsync(id), який отримує сутність скарги з репозиторію, а потім за допомогою Mapper перетворюється на ComplaintDto.
- public async Task<PagedResult<ComplaintDto>> GetPageAsync (ComplaintPageRequest pageRequest) – метод що асинхронно повертає сторінку зі скаргами.
- public async Task<List<ComplaintTypeDto>> GetTypesAsync() – метод який асинхронно повертає перелік типів скарг. Виконує запит до репозиторію типів скарг (ComplaintTypeRepository) та перетворює результати на список ComplaintTypeDto.
- public async Task<Result> UpdateAsync(ComplaintSaveDto dto) – метод що асинхронно оновлює існуючу скаргу. Приймає об'єкт типу ComplaintSaveDto, який містить дані для оновлення скарги. Метод отримує сутність скарги через виклик методу GetEntityAsync(dto.Id), виконує валідацію даних, оновлює властивості скарги згідно з переданими даними, зберігає зміни в базі даних через UnitOfWork.SaveChangesAsync(), і повертає оновлену скаргу у вигляді ComplaintD.

- public async Task<Result> ChangeStateAsync (ComplaintChangeStateDto dto) – метод, що асинхронно змінює стан скарги. Приймає об'єкт типу ComplaintChangeStateDto, що містить ідентифікатор скарги та новий стан. Метод отримує сутність скарги через виклик методу GetEntityAsync(dto.Id), виконує валідацію стану скарги, оновлює властивості скарги (стан, дата завершення, відповідальний співробітник, рішення), зберігає зміни у базі даних та повертає оновлену скаргу як ComplaintDto в об'єкті Result.
- protected override async Task<Complaint> GetEntityAsync(int id) – метод, який асинхронно отримує сутність скарги щодо її ідентифікатора (id). Всередині методу виконується запит до репозиторію, включаючи пов'язані сутності (наприклад, EmployeeLock, Product), та повертається знайдена сутність. Якщо сутність не знайдена, викидається ResourceNotFoundException.
- private static string BuildTaskBody(Complaint complaint) (рис 3.8) – метод, який будує текст завдання на основі даних скарги. Метод форматує інформацію про скаргу у певному форматі, використовуючи клас StringBuilder, та повертає згенерований текст завдання.

```
private static string BuildTaskBody(Complaint complaint)
{
    const string taskItemFormat = "[*][B]{0}: [/B]{1}";

    StringBuilder sb = new StringBuilder();
    sb.Append("[LIST=1]");
    sb.AppendFormat(CultureInfo.InvariantCulture, taskItemFormat, "Источник", DictionaryItemBase.GetById<ComplaintSource>(complaint.SourceId).Name);
    sb.AppendFormat(CultureInfo.InvariantCulture, taskItemFormat, "Тип", complaint.Type.Name);
    sb.AppendFormat(CultureInfo.InvariantCulture, taskItemFormat, "Приоритет", ((Priority)complaint.PriorityId).GetDisplayAttributeName());
    sb.AppendFormat(CultureInfo.InvariantCulture, taskItemFormat, "Описание жалобы", complaint.Text);
    sb.AppendFormat(CultureInfo.InvariantCulture, taskItemFormat, "ФИО", complaint.Fio);
    sb.AppendFormat(CultureInfo.InvariantCulture, taskItemFormat, "Телефон", string.Join(", ", new[] { complaint.Phone, complaint.Phone2 }.Where(x => !string.IsNullOrWhiteSpace(x))));

    if (complaint.ProductId.HasValue)
    {
        sb.AppendFormat(CultureInfo.InvariantCulture, taskItemFormat, "Товар", complaint.Product.NameFullRu);
    }

    if (complaint.OrderId.HasValue)
    {
        sb.AppendFormat(CultureInfo.InvariantCulture, taskItemFormat, "Заказ", complaint.OrderId);
    }

    if (complaint.ServiceRequestId.HasValue)
    {
        sb.AppendFormat(CultureInfo.InvariantCulture, taskItemFormat, "Сервисная заявка", complaint.ServiceRequestId);
    }

    sb.Append("[/LIST]");

    return sb.ToString();
}
```

Рисунок 3.8 –створення тіла скарги

- private IEnumerable<ValidationFailure> CanChangeState(Complaint complaint, params ComplaintState[] states) – метод, що виконує валідацію можливості зміни стану скарги. Приймає сутність скарги та масив допустимих

станів. Метод перевіряє, що скаргу заблоковано поточним користувачем і перебуває в одному із зазначених станів. Якщо перевірка не проходить, метод повертає колекцію ValidationFailure з інформацією про помилки валідації.

- private Complaint Map(ComplaintCreateDto source, Complaint target) – метод, який виконує мапінг даних із сутності Complaint об'єкт ComplaintDto. Метод надає відповідні властивості сутності скарги об'єкту ComplaintDto, такі як ідентифікатор, назва, опис, дата створення, стан, відповідальний співробітник та інші властивості.
- private Complaint Map(ComplaintSaveDto source, Complaint target) – метод, який виконує мапінг даних із сутності ComplaintType в об'єкт ComplaintTypeDto. Метод надає відповідні властивості сутності типу скарги об'єкту ComplaintTypeDto, такі як ідентифікатор, назва та опис.

Контролер «Controller» відповідає за обробку веб-запитів, зв'язаних зі скаргами. Він використовує сервіс IComplaintService, який надає методи для виконання операцій зі скаргами, таких як створення, отримання, оновлення та зміна стану.

У методі CreateAsync контролера відбувається обробка HTTP POST-запиту для створення нової скарги. Він приймає об'єкт ComplaintCreateDto в запиті і передає його в метод CreateAsync сервісу ComplaintService. Результат операції зберігається в змінну результат, а потім повертається у вигляді IActionResult з кодом стану HTTP 200 (Created).

Метод GetOneAsync обробляє HTTP GET-запит для отримання однієї скарги щодо її ідентифікатора. Він приймає ідентифікатор скарги як параметр ID і викликає метод GetOneAsync сервісу ComplaintService. Результат операції зберігається в змінну transferObject типу ComplaintDto, а потім повертається у вигляді JSON-відповіді.

Метод GetAllAsync обробляє HTTP GET-запит для отримання списку скарг із можливістю посторінкової навігації. Він приймає параметри запиту searchCriteria, skip і take, і створює об'єкт ComplaintPageRequest, який містить інформацію про пошукові критерії та налаштування пагінації. Потім метод

викликає GetPageAsync сервісу ComplaintService, передаючи створений об'єкт pageRequest. Результат операції зберігається в змінну pagedResult типу PagedResult<ComplaintDto>, а потім повертається у вигляді JSON-відповіді.

UpdateAsync метод обробляє HTTP PUT-запит для оновлення існуючої скарги. Він приймає ідентифікатор скарги як параметр id і об'єкт ComplaintSaveDto у запиті. Потім метод встановлює ID вхідного об'єкта рівним переданому і викликає метод UpdateAsync сервісу ComplaintService. Результат операції зберігається в змінний результат, а потім повертається у вигляді IActionResult.

Метод GetAllTypesAsync обробляє HTTP GET-запит для отримання списку типів скарг. Він викликає метод GetTypesAsync сервісу ComplaintService, який повертає перелік типів скарг. Результат операції зберігається в змінну типів типу List<ComplaintTypeDto>, а потім повертається у вигляді JSON-відповіді.

Метод TryLockAsync викликає метод TryLockAsync сервісу ComplaintService для спроби блокування скарги. Результат операції зберігається у змінну response типу LockResponse<ComplaintDto>, а потім повертається у вигляді JSON-відповіді.

Метод TryUnlockAsync викликає метод TryUnLockAsync сервісу ComplaintService для спроб розблокування скарги. Результат операції зберігається в змінну lockResponse типу LockResponse<ComplaintDto>, а потім повертається у вигляді JSON-відповіді.

Метод ChangeStateAsync викликає метод ChangeStateAsync сервісу ComplaintService для зміни стану скарги. Він також встановлює ID вхідного об'єкта ComplaintChangeStateDto рівним переданому ідентифікатору. Результат операції зберігається в змінний результат, а потім повертається у вигляді IActionResult.

Всі методи контролера асинхронні та використовують ключове слово await для очікування виконання асинхронних операцій.

Клас Controller надає API-інтерфейс для керування скаргами, дозволяючи створювати, отримувати, оновлювати та змінювати стан скарг. Він застосовує

атрибути для маршрутизації запитів, авторизації та вказівки формату відповідей. Взаємодія з функціональністю скарг здійснюється через сервіс IComplaintService, який інкапсулює бізнес-логіку та операції з даними скарг.

3.4 Тестування

Створимо скаргу, та перевіримо роботу модуля. Користувачу необхідно відкрити модуль в системі (рис 3.9)

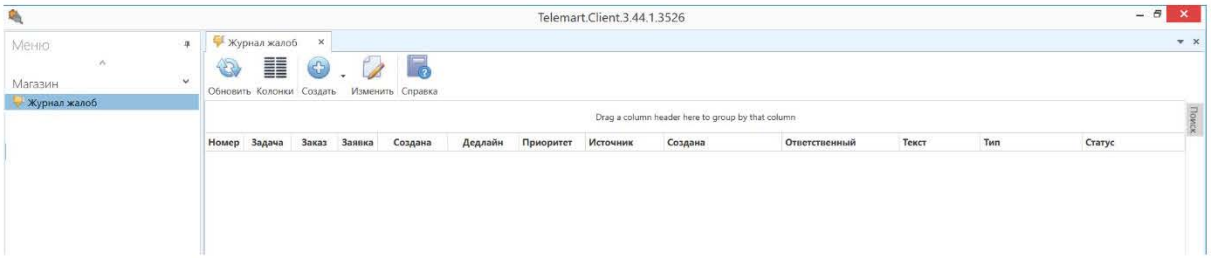

Рисунок 3.9 – форма «Журнал скарг»

Необхідно обрати джерело звернення (рис 3.10), обираємо «Інше»

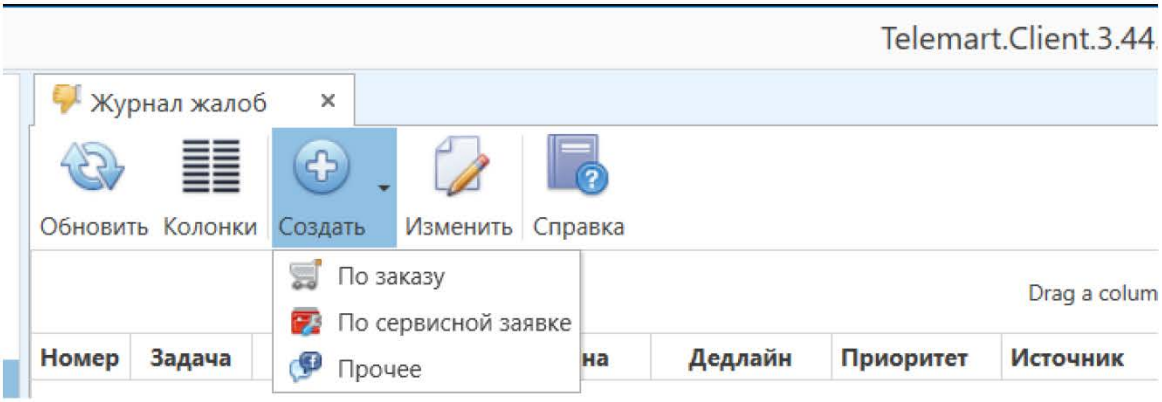

Рисунок 3.10 – вибір джерела звернення

Відкривається форма створення скарги (рис 3.11), у якій необхідно заповнити усі поля:

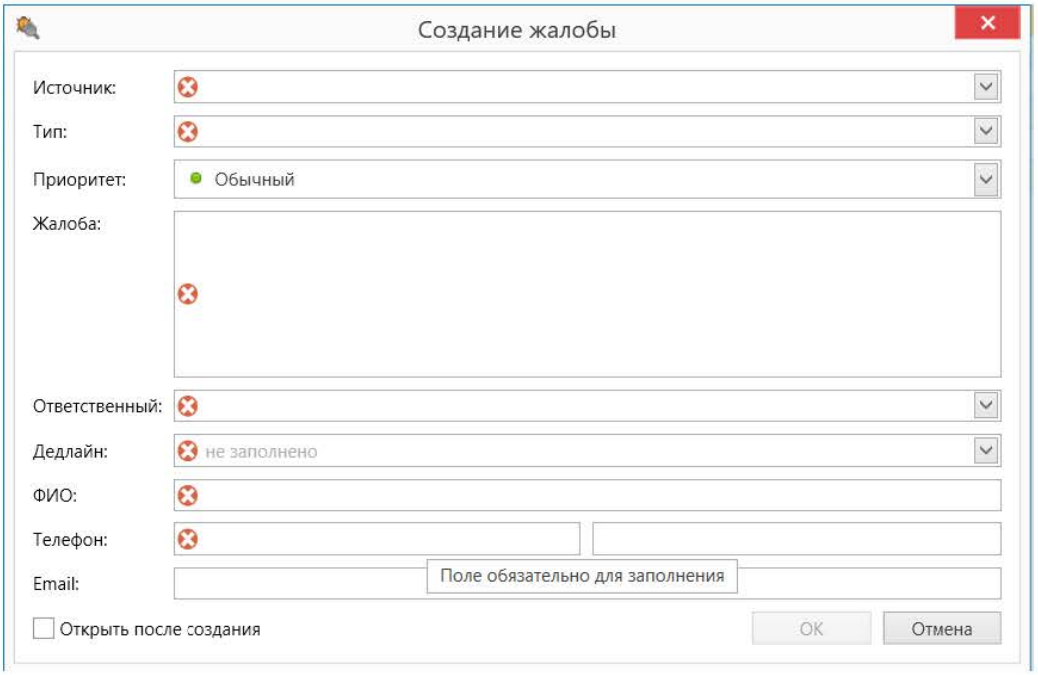

Рисунок 3.11 – форма «створення скарги»

Обираємо джерело звернення (рис 3.12)

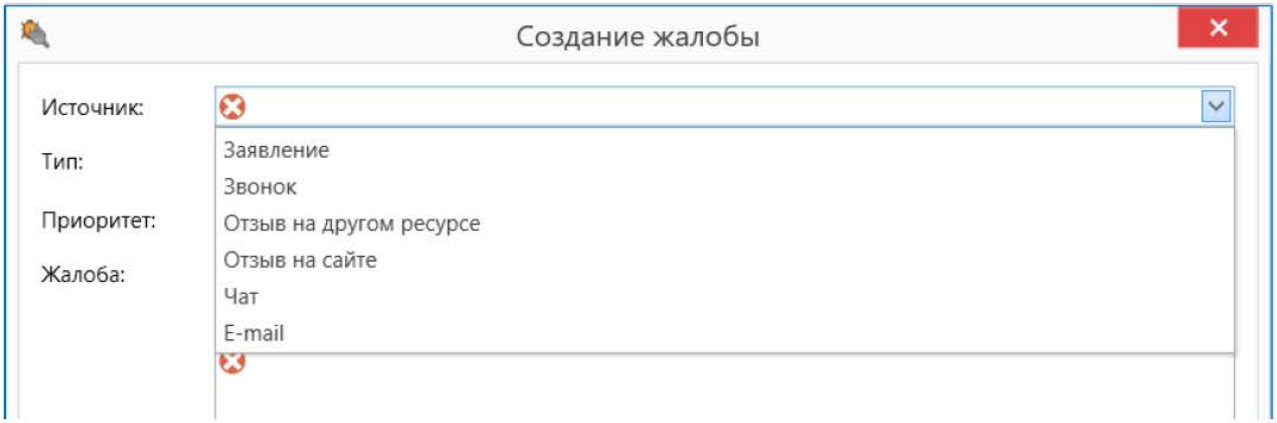

Рисунок 3.12 – Обираємо джерело звернення

Обираємо «Тип жалоби» (рис 3.13). Є список, з якого необхідно обрати з чи саме пов’язана скарга.

Рисунок 3.13 – тип жалоби

Заповнюємо інші поля та нажимаємо «Ок» (рис 3.14)

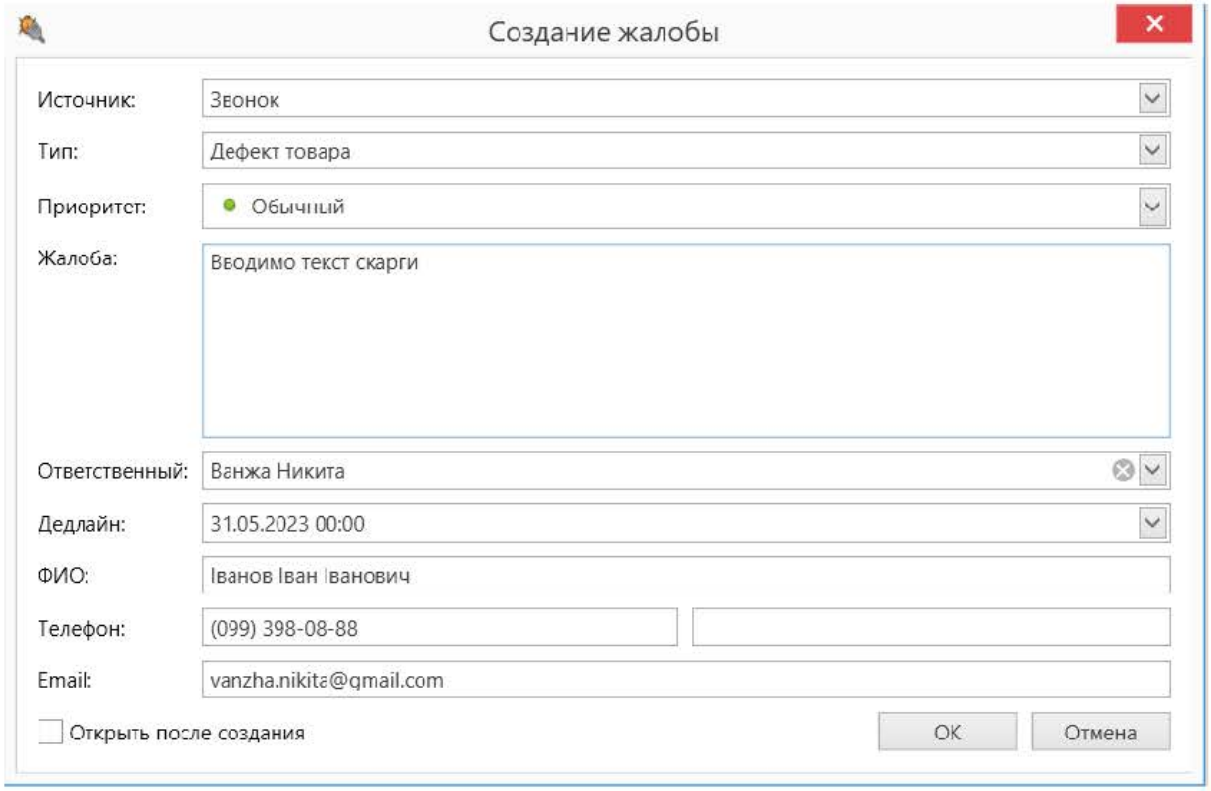

Рисунок 3.14 – заповнення всіх полів форми

В поле «Відповідальний» виводяться усі співробітники компанії, в ньому необхідно обрати відповідального співробітника. Надалі скарга зберігається в системі (рис 3.15) та створюється завдання в CRM системі Bitrix (рис 3.16). Якщо

поставити галочку напроти «Открыть после создания», відкривається Bitrix в браузері з цим завданням.

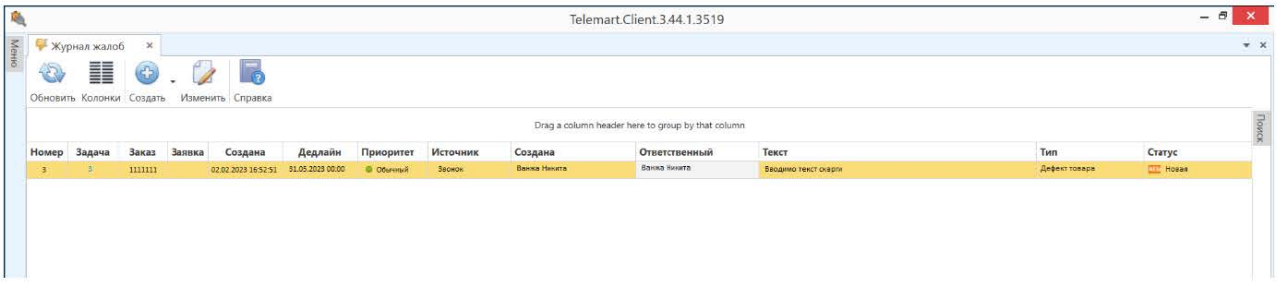

Рисунок 3.15 – створена скарга в системі

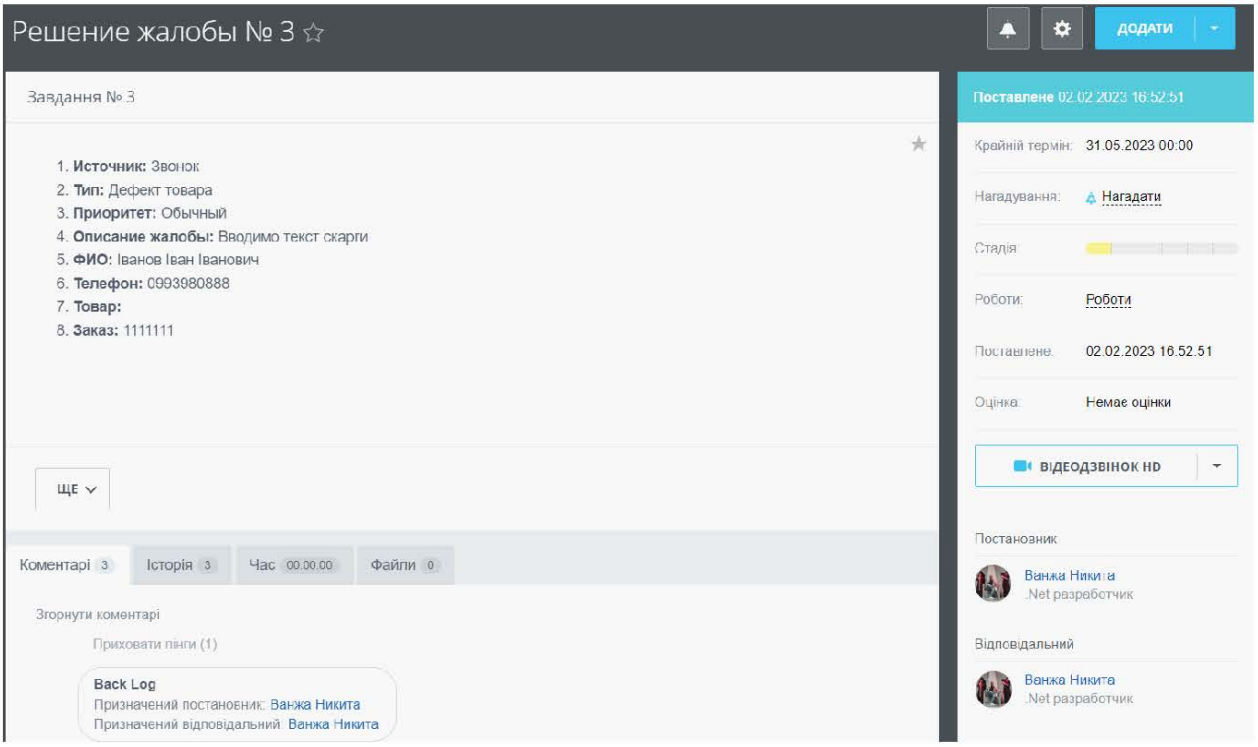

Рисунок 3.16 – завдання в CRM

Якщо необхідно продивитись скаргу, необхідно 2 рази натиснути ЛКМ по скарзі в модулі, та відкриється форма (рис 3.17) для перегляду чи редагування скарги.

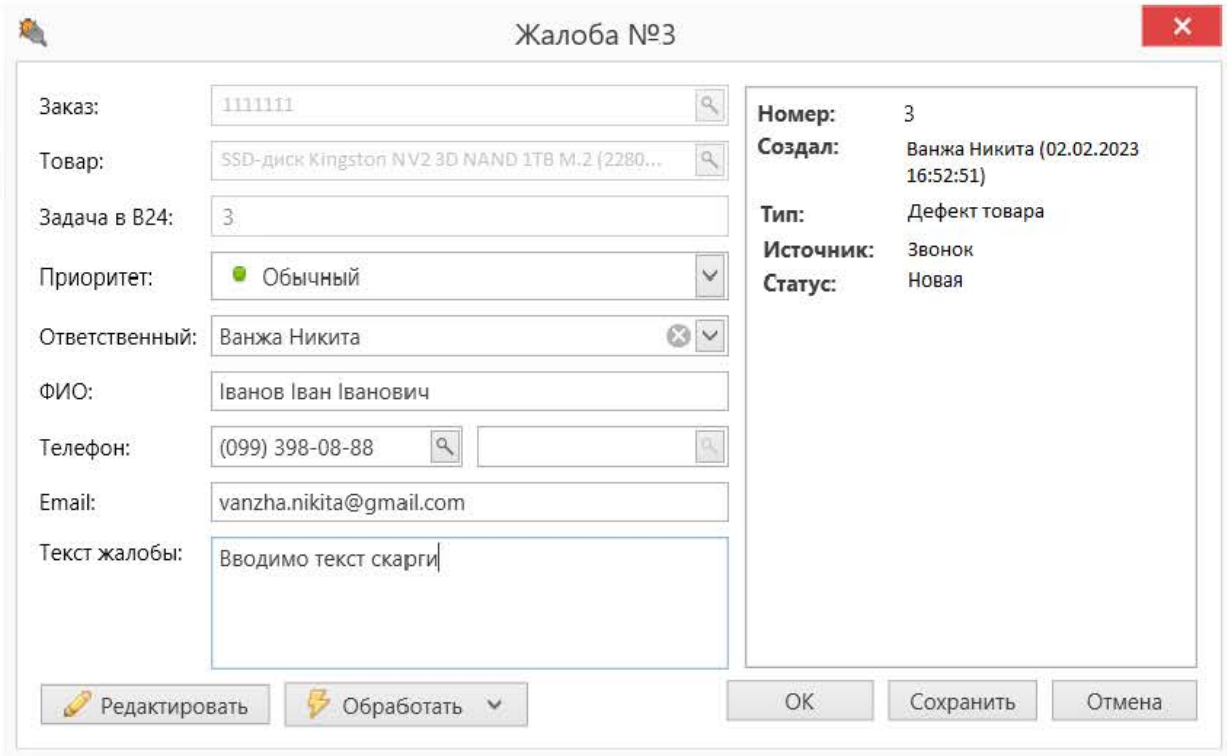

Рисунок 3.17 – перегляд/редагування скарги

Якщо необхідно знайти скаргу, то відкриваємо «Поиск» та заповнюємо поля, по яким необхідно знайти скаргу (рис 3.18).

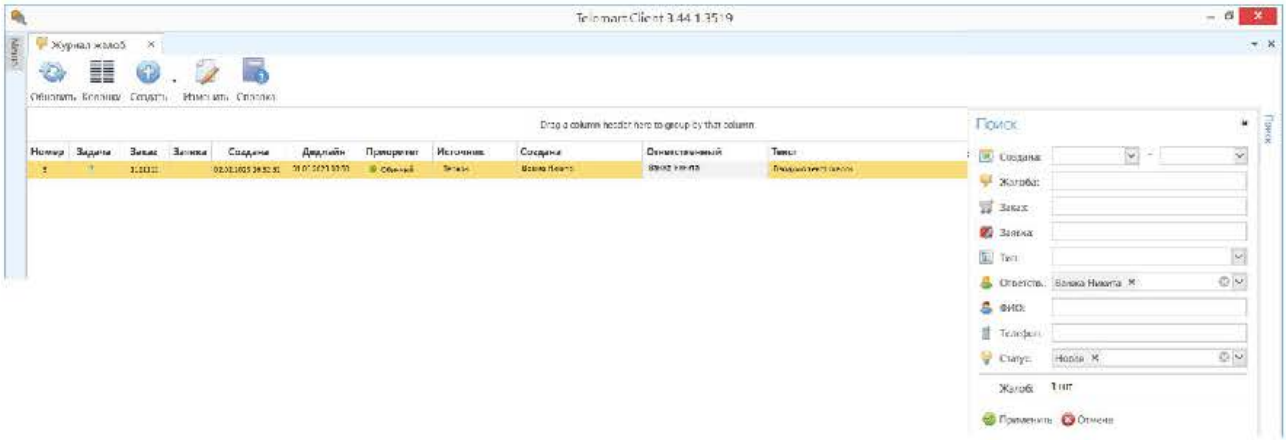

Рисунок 3.18 – вікно для пошуку необхідної скарги

3.5 Висновки до третього розділу

В результаті розробки журналу скарг було створено такі компоненти: основне вікно "Журнал скарг", форма створення скарги, форма перегляду/редагування скарги та вікно пошуку скарг.

Основне вікно "Журнал скарг" є центральною частиною програми, де відображаються всі записи про скарги. Воно містить поля, вказані в технічному завданні, і дозволяє редагувати ці поля за допомогою редактора стовпців. Це основне вікно, яке користувач бачить під час запуску програми та де здійснюється основна робота зі скаргами.

Форма створення скарги надає користувачеві можливість додати нову скаргу до журналу. У цій формі користувач вводить відповідні дані щодо скарги, такі як тип скарги, опис, дата та інші відомості. Після заповнення всіх полів, користувач може зберегти нову скаргу в системі.

Форма перегляду/редагування скарги призначена для перегляду та зміни існуючих записів про скарги. Форма дозволяє вносити зміни до інформації про скаргу, зберігати зміни та оновлювати дані в журналі.

Вікно пошуку скарг надає користувачеві інструменти для пошуку конкретних скарг в журналі на основі заданих критеріїв. Користувач може вказати параметри пошуку, такі як тип скарги, дата, ключові слова та інші фільтри, щоб знайти потрібну скаргу.

Під час розробки журналу скарг було створено такі класи: ClientViewModel, Controller та Service.

ClientViewModel: Цей клас представляє модель подання клієнта у журналі скарг. Він містить властивості, що відображають дані клієнта, такі як ім'я, контактна інформація та інші відповідні поля. ClientViewModel служить передачі даних про клієнта між поданням і контролером.

Controller: Цей клас представляє контролер у журналі скарг. Контролер обробляє запити від клієнта та виконує відповідні дії, такі як створення,

редагування та пошук скарг. У контролері визначено методи, які приймають вхідні параметри від клієнта, взаємодіють із відповідними сервісами та повертають результати назад клієнту. Контролер служить для координації роботи між представленням та сервісом.

Service: Цей клас представляє сервіс у журналі скарг. Сервіс містить бізнес-логіку та операції, пов'язані зі скаргами. Він забезпечує функціональність, таку як створення, читання, та оновлення, а також пошук за заданими критеріями. Сервіс відповідає за обробку бізнес-логіки та забезпечення взаємодії з базою даних або іншими джерелами даних.

Загалом розроблені компоненти та класи забезпечують повний функціонал журналу скарг. Основне вікно надає огляд усіх скарг, форма створення дозволяє додавати нові записи, форма перегляду/редагування забезпечує можливість перегляду та зміни існуючих записів, а вікно для пошуку допомагає користувачеві знаходити потрібні скарги за допомогою заданих фільтрів.

ВИСНОВКИ

В процесі написання кваліфікаційної роботи було встановлено, що розробка модуля є складним завданням, яке потребує детального аналізу, проектування та розробки. Він дозволяє швидко та ефективно отримувати та класифікувати повідомлення від клієнтів з різних каналів комунікації, спрощує процес обробки та вирішення проблем, а також надає інструменти для аналізу та використання пропозицій клієнтів. Його реалізація має ряд переваг і може сприяти покращенню бізнес-процесів інтернет магазину.

Однією з головних переваг розробленого модуля є зручність та швидкість реагування на скарги та пропозиції клієнтів. Завдяки йому, клієнти можуть легко залишити скарги на надані послуги або продукти, що забезпечує їм зручність та можливість ефективно спілкуватися з інтернет-магазином. Це сприяє збереженню лояльності клієнтів та задоволення від обслуговування.

Крім того, модуль "Журнал скарг" забезпечує збір даних про скарги клієнтів. Це дає змогу інтернет магазину отримати уявлення про проблемні аспекти своєї роботи та виявити слабкі місця в бізнес-процесах. Аналіз скарг може допомогти виявити поширені проблеми, які потребують уваги та вжиття заходів для їх вирішення. Це може призвести до покращення якості обслуговування та задоволення клієнтів, що в свою чергу позитивно впливає на репутацію інтернет магазину.

Модуль "Журнал скарг" надає можливість автоматичного створення задач для вирішення скарг. Це дозволяє ефективно організувати роботу з вирішення проблем та забезпечити їх своєчасне вирішення. Автоматичне створення задач дозволяє забезпеч ти, що жодна скарга не залишиться без уваги та буде оброблена вчасно і ефективно. Автоматизація процесу обробки повідомлень дозволяє зменшити людські помилки та затримки, що сприяє підвищенню ефективності роботи.

До недоліків можна віднести, що розроблений модуль спеціалізований під потреби конкретного інтернет-магазину. Це може обмежувати його використання в інших компаніях або вимагати додаткових зусиль для його налаштування під інші умови.

Можливість помилок та недоліків: автоматизована система обробки скарг та пропозицій може містити помилки або недоліки. Це може призвести до некоректної обробки скарг або втрати деяких даних. Потрібно забезпечити високу якість тестування та постійне оновлення системи для зменшення ризиків таких проблем.

Однак, переваги використання модуля "Журнал скарг" переважають його недоліки. Завдяки системі фіксації та аналізу скарг клієнтів, інтернет магазин може покращити якість своїх послуг та продуктів, виявити проблемні місця в бізнес-процесах і прийняти відповідні заходи для їх вирішення. Це сприятиме підвищенню рівня задоволення клієнтів, збереженню їх лояльності та покращенню репутації інтернет магазину.

Пропозиції щодо подальшого розвитку та модернізації модуля "Журнал скарг" можуть включати наступні аспекти:

Розширення функціональності: можливість прикріплювати додаткові файли до скарги. Постійне оновлення і поліпшення інтерфейсу модуля "Журнал скарг" для забезпечення зручності та ефективності роботи користувачів. Використання сучасних технологій та дизайну допоможе зробити інтерфейс більш привабливим та інтуїтивно зрозумілим.

Вдосконалення аналітичного модуля: додавання можливості генерації звітів і статистики щодо скарг, виявлення трендів та патернів у скаргах клієнтів, що дозволить знайти корисні інсайти для вдосконалення бізнес-процесів і покращення якості обслуговування.

Зазначені напрямки подальшого розвитку модуля "Журнал скарг" допоможуть інтернет-магазину покращити комунікацію з клієнтами, ефективно вирішувати скарги, виявляти та вирішувати проблеми в роботі бізнесу. Такі

заходи сприятимуть зміцненню взаємовідносин з клієнтами, підвищенню рівня задоволення клієнтів.

Модуль "Журнал скарг" є цінним інструментом для покращення бізнес-процесів інтернет-магазинів. Він дозволяє зручно та швидко фіксувати скарги клієнтів, аналізувати їх та вживати відповідних заходів для вирішення проблем. Застосування модуля "Журнал скарг" може привести до покращення якості обслуговування, збереження лояльності клієнтів та підвищення конкурентоспроможності інтернет-магазину. Його переваги включають зручність взаємодії з клієнтами, можливість збору та аналізу даних про скарги, автоматичне створення задач для вирішення проблем, а також можливість подальшого розширення.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Рязанцев А. Як запровадити CRM-систему за 50 днів / Олексій Рязанцев., 2017.

2. Measuring Your Net Promoter Score | Bain & Company [Електронний ресурс] – Режим доступу до ресурсу: https://www.netpromotersystem.com/about/measuring-your-net-promoter-score/

3. Система CRM – Контроль Бази Клієнтів [Електронний ресурс] – Режим доступу до ресурсу: https://ua.keycrm.app

4. LiveAgent | Simple Customer Support Software for Teams [Електронний ресурс] – Режим доступу до ресурсу: https://www.liveagent.com/pricing/

5. Creatio - CRM-система Creatio [Електронний ресурс] – Режим доступу до ресурсу: https://www.creatio.com/ua/

6. Щенніков, С. Ю. Реінженіринг бізнес процесів, управління, планування та оцінка [Текст] / С. Ю. Щенніков. - М.: Вісь-89, 2004. - 288 с.

7. Бізнес-процеси інтернет-магазину та їх оптимізація [Електронний ресурс] – Режим доступу до ресурсу: https://wezom.com.ua/ua/blog/biznes-processy-internet-magazina-i-ih-optimizaciya

8. Переваги та недоліки .NET: швидкий розвиток, велика поширеність і середні зарплати [Електронний ресурс] – Режим доступу до ресурсу: https://dou.ua/lenta/articles/pros-and-cons-of-dotnet/

9. Introduction to authorization in ASP.NET Core [Електронний ресурс] – Режим доступу до ресурсу: https://learn.microsoft.com/en-us/aspnet/core/security/authorization/introduction?view=aspnetcore-7.0

10. База даних MySQL [Електронний ресурс] – Режим доступу до ресурсу: https://promoter.net.ua/articles/baza-danix-mysql.html

11. SQL в Access: основні поняття, глосарій і синтаксис [Електронний ресурс] – Режим доступу до ресурсу: https://support.microsoft.com/uk-

ua/office/sql-в-access-основні-поняття-глосарій-і-синтаксис-444d0303-cde1-424e-9a74-e8dc3e460671

12. How to use FluentValidation in ASP.NET Core (.NET 6) [Електронний ресурс] – Режим доступу до ресурсу: 11. https://blog.christian-schou.dk/how-to-use-fluentvalidation-in-asp-net-core6/

13. What is AutoMapper? [Електронний ресурс] – Режим доступу до ресурсу: https://automapper.org

14. Загальні відомості про моделі, представлення та контролери (C#) [Електронний ресурс] – Режим доступу до ресурсу: https://learn.microsoft.com/ru-ru/aspnet/mvc/overview/older-versions-1/overview/understanding-models-views-and-controllers-cs