Міністерство освіти і науки України
Національний технічний університет
«Дніпровська політехніка»

інформаційних технологій
(факультет)
Кафедра системного аналізу і управління
(повна назва)

ПОЯСНЮВАЛЬНА ЗАПИСКА
кваліфікаційної роботи ступеню магістра

студента Василенко Юрія Анатолійовича
(ПІБ)
академічної групи 124м–21-1

(шифр)
спеціальності 124 Системний аналіз
(код і назва спеціальності)
спеціалізації[1]
за освітньо-професійною програмою Системний аналіз
(офіційна назва)
на тему «Аналіз методологій створення web–сайтів та CRM–систем»

(назва за наказом ректора)

Керівники	Прізвище, ініціали	Оцінка за шкалою		Підпис
		рейтинговою	інституційною	
кваліфікаційної роботи	Д.т.н., проф. Слєсарєв В. В			
розділів:	2			
Інформаційно-аналітичний розділ	Д.т.н., проф. Слєсарєв В. В			
Спеціальний розділ	Д.т.н., проф. Слєсарєв В. В			

Рецензент				

Нормоконтролер	к.ф.-м.н., доц. Хом'як Т.В			

Дніпро
2022

ЗАТВЕРДЖЕНО:
завідувач кафедри
системного аналізу та управління
(повна назва)
_____ к.т.н., доц. Желдак Т.А.
(підпис) (прізвище, ініціали)

«_____»_____ 2022 року

ЗАВДАННЯ
на кваліфікаційну роботу
ступеня магістра

студенту Василенко Ю. А. **академічної групи** 124м–21
(прізвище та ініціали) (шифр)

Спеціальності 124 Системний аналіз

на тему «Аналіз методологій створення web–сайтів та CRM–систем »

затверджену наказом ректора НТУ «Дніпровська політехніка» від 31.10.2022_№ 1200-с

Розділ	Зміст	Термін виконання
1. Інформаційно-аналітичний	Розгляд інструментів та механізмів створення веб-сайтів, CRM-систем	28.09.2022 – 11.10.2022
2. Спеціальний	Аналіз реального тестового завдання клієнта, використання метода аналізу ієрархій для вибору CRM-системи, створення CRM-системи	12.10.2022 – 29.11.2022

Завдання видано _____ Слєсарєв В.В.
(підпис керівника) (прізвище, ініціали)

Дата видачі 28 вересня 2022 р.

Дата подання до екзаменаційної комісії 1 грудня 2022 р.

Прийнято до виконання _____ Василенко Ю. А.
(підпис студента) (прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 54 с., 4 додатки, 15 джерел, 28 рис., 24 табл.

Актуальність обраної теми обумовлена високою конкуренцією на інтернет-ринку, адже правильно обрані та оптимізовані веб-проекти мають вищу швидкість відповіді на запит клієнта.

Об'єкт досліджень – процес вибору та розробки CRM-системи.

Предмет досліджень – CRM-система роботи з клієнтами агро-бізнесу.

Мета роботи – запропонувати шляхи та методи покращення економічних показників підприємства в ході впровадження CRM-системи.

Методи дослідження: Аналіз методів автоматизації та стандартизації управління відносин з клієнтами, механізми впровадження CRM-системи.

Поставлені задачі:

1) Вибір веб-додатка для покращення економічних показників підприємства.
2) Розгляд існуючих CRM-систем та розгляд ідеї розробки приватної системи.
3) Розробка приватної CRM-системи за тестовим завданням реального клієнта.

В інформаційно–аналітичному розділі наведено аналіз об'єкту дослідження та ключових проблем на ньому. Поставлені задачі дослідження та обрано концепції їх розв'язання.

У спеціальному розділі розроблене тестове завдання для CRM – системи та використаний метод ієрархій для вибору CRM – системи

Практична цінність отриманих в роботі результатів полягає в тому, що впровадження сучасних веб-додатків для роботи з клієнтами спричиняє підвищення економічних показників для власника.

Ключові слова: ВЕБ-САЙТ, ВЕБ-ДОДАТОК, САЙТ, АЛЬТЕРНАТИВА, ВИБІР, CRM-СИСТЕМА, СЕРВЕР, КЛІЄНТ, ЗАПИТ, РОЗРОБКА, ПРОГРАМУВАННЯ, РЕАЛІЗАЦІЯ, НАВАНТАЖЕННЯ.

ABSTRACT

Explanatory note: 54 pages., 4 applications, 15 sources, 28 pic., 24 tables.

The relevance of the chosen topic is due to high competition in the Internet market, because properly selected and optimized web projects have a higher speed of response to customer requests.

The object of research is the process of selection and development of CRM-system.

The subject of research is CRM-system of work with agribusiness clients.

The purpose of the work is to offer ways and methods to improve the economic performance of the enterprise during the implementation of the CRM-system.

The informational and analytical section provides an analysis of the research object and its key problems. Research tasks are set and concepts for their solution are chosen.

In a special section, a test task for the CRM system was developed and the hierarchy method was used to select a CRM system

Tasks:

1) Choosing a web application to improve the economic performance of the enterprise.
2) Consideration of existing CRM-systems and consideration of the idea of developing a private system.
3) Development of a private CRM-system for a test task of a real client.

The practical value of the results obtained in the work is that the introduction of modern web applications for working with customers leads to increased economic performance for the owner. It is important to properly analyze the client's needs and use the appropriate tools.

Keywords: WEBSITE, WEB APP, SITE, ALTERNATIVE, CHOICE, CRM SYSTEM, SERVER, CLIENT, REQUEST, DEVELOPMENT, PROGRAMMING, LOAD.

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАЧЕННЯ

CRM (*Customer relationship management*) – налагоджена система інструментів ведення бізнесу. Замість табличок Excel, месенджерів, багатьох документів та біганини по кабінетах залишається один-єдиний веб-сервіс.

Web – система доступу до пов'язаних між собою документів на різних комп'ютерах, підключених до Інтернету.

HTML (*HyperText Markup Language*) – мова розмітки гіпертекстових документів.

CSS (*Cascading Style Sheets*) - каскадні таблиці стилів.

PHP (*Hypertext Preprocessor*) – мова програмування, гіпертекстовий препроцесор.

БД – база даних.

Фреймворк (англ. *Framework, структура*) – інфраструктура програмних рішень, що полегшує розробку складних систем.

MAI – метод аналіза ієрархій.

MVC (англ. *Model View Controller*) – модель-уявлення-контролер.

CRUD – create, read, update, delete – основні функції роботи з базою даних.

ВСТУП

В даний час впровадження інновацій дуже важливо не тільки для загального зростання конкурентоспроможності підприємств, а й для формування ефективних клієнтських відносин, що забезпечують, в свою чергу, прибутковість компанії. І сьогодні більшість керівників компаній почали розуміти, що одна оптимізація виробництва вже не вирішує проблему виживання. Особливо це помітно в сфері послуг, де компанії залежать не стільки від якості самих продуктів або послуг, скільки від досконалості механізмів взаємодії компанії зі своїми клієнтами. У тих сферах, де наростання конкуренції йде вельми активно поряд з швидким оновленням технологій і продукції, включається ще один зовнішній фактор – інвестиційний. Для виведення нових товарів і послуг на масовий ринок компаніям вже недостатньо власних коштів. Залучення ж зовнішніх інвестицій зазвичай супроводжується вимогою якнайшвидшого їх повернення, що прямо визначає необхідну швидкість нарощування доходів і, отже, клієнтської бази. [1]

Сучасний ринок пропонує нам безмежно широкі можливості для вибору товарів або послуг. У поштову скриньку регулярно приходять повідомлення про акції і знижки, на телефон надходять інформаційні дзвінки з пропозиціями скористатися послугами тієї чи іншої компанії, в Інтернеті регулярно з'являється наздоганяюча реклама з пропозицією купити товар. Проблема в тому, що всі товари і послуги зараз в цілому приблизно однакові. Істотні відмінності знайти досить важко, часто вони і зовсім не хвилюють клієнта.

Конкурентна боротьба в таких умовах стає особливо гострою. Залучення нових клієнтів коштує грошей, але після того як вони залучені їх потрібно ще й утримати. Потрібно сподобатися клієнту і вибудувати з ним успішні і довгострокові відносини, або зробити так, щоб він як мінімум не залишився незадоволеним після звернення в компанію або здійснення

покупки. Один незадоволений клієнт може позбавити компанію ще десятка потенційних клієнтів. Як вибудувати добрі стосунки з клієнтом? Як утримати його і не дати піти до конкурентів? Як прихилити його до повторної покупки?

Вирішити ці питання допоможе впровадження CRM-системи. Відразу варто уточнити, саме по собі впровадження не стане чарівною паличкою, за помахом якої клієнти відразу стануть задоволеними і відданими вашій компанії в цілому або продукту зокрема. CRM - це інструмент, майстерністю володіння яким потрібно оволодіти. Якщо говорити про сучасні CRM, то це набір інструментів, який допомагає бізнесу вибудовувати відносини з клієнтами. [2]

При відсутності CRM-системи, кожен співробітник веде свою клієнтську базу так як йому хочеться. Вона може зберігається у вигляді номерів в телефоні, в блокноті або в таблицях. Справи плануються в щоденнику. Такі способи ведення клієнтської бази можуть бути ефективні тільки в тому випадку якщо ефективний сам співробітник. Якщо він дійсно після звернення клієнта занесе його в свою базу, не забуде відправити йому пропозицію і передзвонить йому. [3]

1 ІНФОРМАЦІЙНО-АНАЛІТИЧНИЙ РОЗДІЛ

1.1 Як працює WEB?

Комп'ютери, приєднані до мережі, називаються *клієнтами* та *серверами*. Цей малюнок спрощено показує, як відбувається їхня взаємодія (рис 1.1):

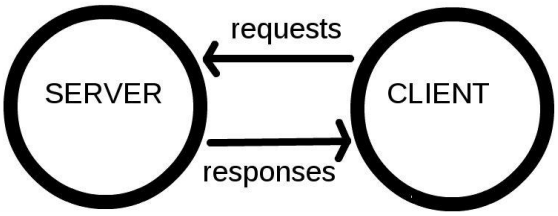

Рис. 1.1. Взаємодія клієнта та сервера.

Клієнти — це звичайні користувацькі комп'ютери, під'єднані до інтернету (наприклад, ваш комп'ютер, під'єднаний до Wi-Fi, або ваш телефон, під'єднаний до мобільної мережі) і програмне забезпечення на цих комп'ютерах з веб-доступом (зазвичай, це браузери, такі як Firefox чи Chrome).

Сервери — це комп'ютери, які зберігають веб-сторінки, сайти чи додатки. Коли клієнт хочу отримати доступ до веб-сторінки, копія цієї сторінки завантажується із сервера на клієнт і відображається у веб-браузері клієнта.

Уявіть, що коли ви набираєте https://some-persons.com, ви відправляєте листа, що говорить: "Шановні some persons, я хочу побачити веб-сайт some-persons.com. Відправте мені його, будь ласка!"

Ваш лист надійде до найближчого поштового відділення. Далі він надходить до іншого, що знаходиться трохи ближче до вашого адресата, далі до іншого і так далі, поки не буде досягнуто остаточного пункту

призначення. Єдина відмінність в тому, що при відправці багатьох листів (пакетів даних) в однакове місце, кожен з них може піти через абсолютно різні поштові відділення (маршрутизатори). Це залежить від того, як їх розподілили в кожному відділенні.

Так, це так само просто. Ви відправляєте повідомлення і очікуєте певної відповіді. Звісно, замість паперу і ручки ви використовуєте байти даних, але ідея та сама!

Замість адрес із назвами вулиці, міста, індексом і назвою країни, ми використовуємо IP адреси. Ваш комп'ютер спочатку запитує про DNS (англ. Domain Name System - система доменних імен) для переводу djangogirls.org в IP-адресу. Це працює подібно до старомодних телефонних довідників, де ви могли б знайти ім'я особи, з якою бажаєте зв'язатися і знайти її номер телефону та адресу.

Коли ви відправляєте листа, це потребує наявності певних речей для коректної доставки: адреса, штамп тощо. Також ви використовуєте мову, котру розуміє отримувач, правда ж? Те ж саме стосується і пакетів даних, які ви відправляєте, щоб побачити веб-сайт. Ми використовуємо протокол, який називається HTTP (англ. Hypertext Transfer Protocol, протокол передачі гіпертекстових документів).

Отже, по суті, коли у вас є веб-сайт, вам треба мати сервер (машину), де він житиме. Коли сервер отримує вхідний запит (в листі), він відправляє назад ваш веб-сайт (в іншому листі).

1.2 Мова розмітки веб-сторінки HTML

HTML (англ. HyperText Markup Language — мова розмітки гіпертекстових документів) — стандартна мова розмітки веб-сторінок в Інтернеті. Більшість веб-сторінок створюються за допомогою мови HTML (або XHTML). Документ HTML оброблюється браузером та відтворюється на екрані у звичному для людини вигляді. У більшості випадків автор

документа суворо визначає зовнішній вигляд документа. У разі HTML читач, ґрунтуючись на можливостях веб-браузера, може, певною мірою, керувати зовнішнім виглядом документа (але не його вмістом). HTML дозволяє відзначити, де в документі повинен бути заголовок або абзац за допомогою тега HTML, а потім надає веб-браузеру інтерпретувати ці теги. Наприклад, один веб-браузер може розпізнавати тег початку абзацу і представляти документ у потрібному вигляді, а інший не має такої можливості і 13 надає документ в один рядок. Користувачі деяких Web-браузерів мають, також, можливість налаштовувати розмір і вид шрифту, колір та інші параметри, що впливають на відображення документа. [4]

Документ HTML складається з трьох частин: Декларація типу документа (англ. Document type declaration, Doctype), на початку документа, в якій визначається тип документа (DTD). Шапка документа (знаходиться в межах елемента head), в якій записано загальні технічні відомості або додаткова інформація про документ, яка не відтворюється безпосередньо в браузері; Тіло документа (може знаходитися в елементі body), в якому міститься основна інформація документа. Нижче наведено приклад загальної структури HTML-документа (табл. 1.1, рис. 1.2):

Таблиця 1.1

Приклади відтворення в браузері HTML-коду

Фрагмент HTML-розмітки документа	Відтворення в браузері
<! DOCTYPE html> <html> <head> <title>Назва</title> </head> <body> Давньогрецькі боги Посейдон володар світових вод, Океану, в латинян йому відповідав Нептун, у слов'ян — Цар Моря, Цар Морський, Водяник. </body>	**Давньогрецькі боги** **Посейдон** — володар світових вод, Океану, в латинян йому відповідав *Нептун*, у слов'ян — *Цар Моря, Цар Морський, Водяник*.

`</html>`	

Продовження табл. 1.1

`<a href="https://www.nmu.org.ua/ua/">`
`Національний ТУ «Дніпровська політехніка»`
`</a>`
	Національний ТУ «Дніпровська політехніка»

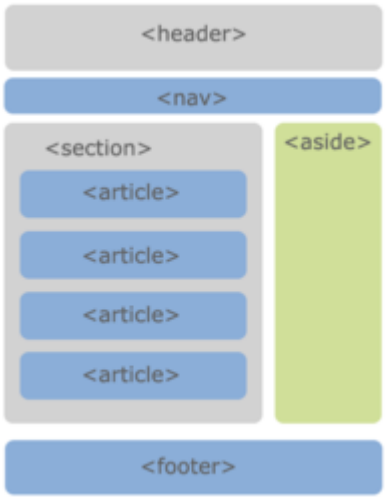

Рис. 1.2. Семантична розмітка сторінки

HTML-документи можуть бути створені за допомогою будь-якого текстового редактора або спеціалізованих HTML-редакторів і конвертерів. Вибір редактора, який буде використовуватися для створення HTML-документів, залежить виключно від поняття зручності і особистих пристрастей кожного автора.

Основна перевага HTML полягає в тому, що ваш документ може бути переглянутий на веб-браузерах різних типів і на різних платформах. Каскадні таблиці стилів (англ. Cascading Style Sheets або скорочено CSS) — спеціальна мова, що використовується для опису сторінок, написаних мовами розмітки даних. Найчастіше CSS використовують для візуальної презентації сторінок, написаних HTML та XHTML, але формат CSS може застосовуватися до

інших видів XML-документів. Таблицю стилів CSS можна вмонтувати прямо в HTML -сторінку - це внутрішня таблиця стилів. Або ж її можна створити в окремому файлі, і вже потім приєднати посилання на нього до потрібної HTML -сторінки - це зовнішня 14 таблиця стилів. Зовнішню таблицю необхідно підключити до основного HTML - документу за допомогою спеціальних тегів:

```
<link rel="stylesheet" type="text/css" href="/style.css>
```

у шапці html-документа, де *style.css* - це ім'я файлу, що містить «таблицю» CSS (рис. 1.4.). Завдяки цьому, стиль, описаний у зовнішній таблиці CSS, можна використовувати повторно скільки завгодно разів. [5]

Основою структури CSS є правило, за яким об’єктам HTML присвоюються стилі (рис. 1.3):

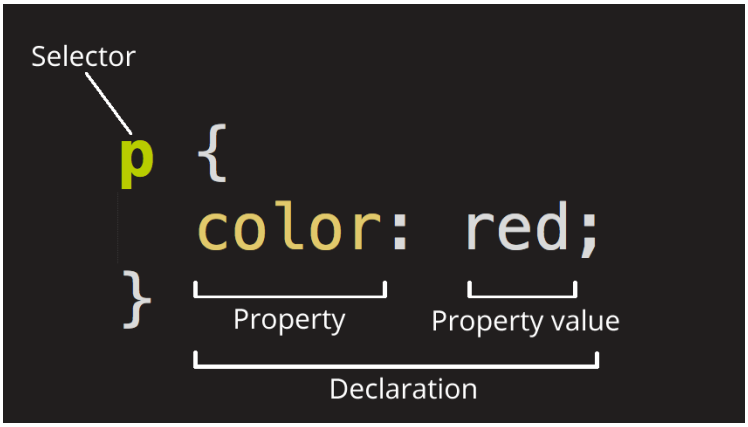

Рис. 1.3. Правило CSS

Тут:

- **p** – елемент, якому надається стиль,
- **Color –** властивість,
- **Red –** значення властивості,
- **Declaration** – визначення (у дужках стилі, які надаються елементу).

Читається як: «*Надати властивість “колір: червоний” елементу p*».

Отже, щоб створити веб-сторінку, достатньо створити за шаблоном html-файл, описати в ньому контент, підключити таблиці стилів css, та описати їх (рис. 1.4 - 1.5) [6]:

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>My test page</title>
    <link href="http://fonts.googleapis.com/css?family=Open+Sans" rel="stylesheet" type="text/css">
    <link href="styles/style.css" rel="stylesheet" type="text/css">
  </head>
  <body>
    <h1>Mozilla is cool</h1>
    <img src="images/firefox-icon.png" alt="The Firefox logo: a flaming fox surrounding the Earth.">

    <p>At Mozilla, we’re a global community of</p>

    <ul> <!-- changed to list in the tutorial -->
      <li>technologists</li>
      <li>thinkers</li>
      <li>builders</li>
    </ul>

    <p>working together to keep the Internet alive and accessible, so people worldwide can be informed contributors and creators of the Web. We believe this act of human collaboration across an open platform is essential to individual growth and our collective future.</p>

    <p>Read the <a href="https://www.mozilla.org/en-US/about/manifesto/">Mozilla Manifesto</a> to learn even more about the values and principles that guide the pursuit of our mission.</p>
  </body>
</html>
```

Рис. 1.4. Приклад html-коду для створення реальної сторінки

```
html {
  font-size: 10px;
  font-family: 'Open Sans', sans-serif;
}


h1 {
  font-size: 60px;
  text-align: center;
}

p, li {
  font-size: 16px;
  line-height: 2;
  letter-spacing: 1px;
}


html {
  background-color: #00539F;
}

body {
  width: 600px;
  margin: 0 auto;
  background-color: #FF9500;
  padding: 0 20px 20px 20px;
  border: 5px solid black;
}

h1 {
```

```
32.   margin: 0;
33.   padding: 20px 0;
34.   color: #00539F;
35.   text-shadow: 3px 3px 1px black;
36. }
37.
38. img {
39.   display: block;
40.   margin: 0 auto;
41. }
```

Рис. 1.5. Приклад таблиці стилів CSS

Таким чином у браузері ми отримаємо наступний результат у браузері (рис. 1.5):

Рис. 1.6. Результат html+css коду у браузері

1.3 Існуючі проблеми в HTML та методи їх вирішення

Взагалі, HTML не є мовою програмування. Це мова розмітки веб-сторінки, тому можна виділити цілу низку обмежень, у порівнянні з іншими мовами програмування. Ви не зможете написати багатофункціональний веб-сайт на мові HTML. Так, ви зможете створити блог, використовуючи html+css, зробити інформаційний сайт, але кожний раз вам доведеться «залазити під капот» вашого сайту, розкривати вихідні файли проекту на хостингу, вносити свої правки і відправляти сайт «в роботу».

Для того щоб користуватися сайтом і використовувати безліч динамічних властивостей, таких як:

- створення,
- редагування,
- видалення,
- перегляд інформації або записів,
- коментування,
- авторизація,
- ролі користувачів та інше,

були придумані динамічні мови програмування, які ми розглянемо далі.

1.4 Динамічні мови програмування: PHP

PHP (від англ. *Hypertext Preprocessor*) - мова програмування, яка спочатку створювалась для розробки веб-додатків, але еволюціонувала в мову загального призначення (рис. 1.7).

PHP відноситься до мов загального призначення. Проте ця мова перш за все застосовується для розробки веб-додатків. Це серверна мова, тобто вона виконується на сервері. Написані на PHP програми отримують дані від

користувачів сайту, обробляють їх, взаємодіють з базами даних, повертають на сайт оброблену інформацію.

Рис. 1.7. Логотип мови програмування PHP

PHP – частково об'єктно-орієнтована мова. Це дає можливість повторно використовувати код, що економить час і сили в процесі розробки. Існує безліч фреймворків PHP: Symfony, CodeIgniter, Laravel, Joomla, WordPress та ін. Кожен з них має свій функціоналом і заточений під певні завдання. PHP спочатку був оптимізований під швидке створення веб-додатків. Йому притаманні такі вбудовані функції, як використання запитів GET і POST, робота з HTML і URL. Для бізнесу це означає, що *час (а, відповідно, і витрати) на розробку скорочуються*, а інвестиції починають *окупатися швидше* [7].

До речі, PHP імплементує HTML, тобто можна додати до php-коду html-розмітку і це не буде помилкою. Або, навпаки, до html-файлу додати php-код, але у такому випадку потрібно буде змінити формат файлу на *.php*.

Рис. 1.8. Приклад додавання php-коду до html

1.5 PHP-фреймворки та CRM-системи

Коли люди говорять про PHP framework, найчастіше спливає назву **Laravel** (англ). Ця специфічна структура відома своїм елегантним синтаксисом, який легко зрозуміти і з яким приємно працювати (рис. 1.9).

З Lavarel ви можете швидко приступити до роботи над своїми проектами. Ви також зможете пропустити багато основ, оскільки ви отримуєте доступ до таких функцій, як аутентифікація користувачів, управління сеансами і кешування. В цілому, Laravel володіє всіма функціональними можливостями, які вам знадобляться для створення сучасного PHP-додатки, що говорить багато про що.

Рис. 1.9. Найпопулярніший php-фреймворк – Laravel.

Якби існував конкурс на кращу середу PHP, **Symfony** (англ) був би там з CodeIgniter і Laravel з точки зору популярності. Як тільки ви починаєте копатися в фреймворку, стає легко зрозуміти, чому. Symfony володіє не тільки крутим ім'ям, але і є дуже гнучким. Він включає в себе пакетну і компонентну систему, яка дозволяє вам вибирати потрібні функції PHP або просто використовувати всю інфраструктуру [8].

Важливо сказати, що будь-який сайт будується на двох «слонах»: *front-end* і *back-end*.

Front-end - клієнтська сторона призначеного для користувача інтерфейсу до програмно-апаратної частини сервісу або, простіше кажучи, то що бачить користувач.

Back-end - програмно-апаратна частина сервісу або «капот» веб-сайту, який відповідає здебільшого за функціональну логіку і зберігання даних. [9]

Взагалі, говорячи про сучасні технології сайтобудування, розробники називають «стеки» розробки. Більшість сучасних сайтів побудовані на декількох мовах програмування та фреймворках. Так, наприклад, Facebook написаний на *JS (front-end), PHP, Hack, Python, C++* і ще кількох. Для зберігання будь-яких даних сайту використовуються бази даних такі як *MySQL, MariaDB, Microsoft SQL Server, PostgreSQL* та інші. База даних відноситься до серверної частини сайту - back-end.

Концепція **CRM** (Customer Relationship Management) означає, що розрізнені інструменти ведення бізнесу об'єднуються в налагоджену систему. Замість табличок Excel, месенджерів, багатьох документів та біганини по кабінетах залишається один-єдиний сервіс. У нього входять програми для збору даних про клієнтів, управління угодами, контролю за менеджерами, аналітики і прогнозування. Він спрощує рутину, прискорює прийняття правильних рішень і виключає помилки.

Як це виглядає в CRM? Уявіть таблицю Excel з вашої клієнтською базою, але тільки при натисканні на ім'я клієнта відкривається зручна картка, в якій міститься вся хронологія роботи з ним - від першого дзвінка до покупки. Тут можна прослухати дзвінки, подивитися історію покупок, створити документи за шаблоном, написати e-mail або sms, поставити задачу.

Коли клієнт дзвонить вам, CRM пропонує відкрити його картку, і ви відразу вітаєте його по імені. Навіть якщо раніше цього покупця вів інший менеджер, ви легко відповісте на його питання без всяких «уточню і передзвоню». CRM сама відправить клієнту sms-повідомлення про статус замовлення і нагадає про зустріч. У підсумку ви економите час - і своє, і клієнта. А значить, робите його більш лояльним і налаштованим на покупку.

«Серцем» будь-якої CRM-системи є база даних фізичних та юридичних осіб, які взаємодіють із вашою компанією в рамках діяльності підприємства. Це не лише клієнти, але і філії компанії, партнери, постачальники, конкуренти. Інформація про клієнтів сама по собі є цінним активом, а грамотне управління даними в системі дозволяє використовувати її в роботі

максимально ефективно. Клієнтська база консолідована, організація отримує повну інформацію про своїх клієнтів та їхніх перевагах і, ґрунтуючись на цих відомостях, будує стратегію взаємодії.

Єдина **база клієнтів** і повна історія відносин із ними в сукупності з потужними аналітичними інструментами CRM дозволяє утримувати і розвивати наявних клієнтів, виявляючи найцінніших, а також залучати нових клієнтів. [10]

1.6 Висновки до розділу та постановка задачі

Складність розуміння та встигання за розвитком галузі має свої плюси та мінуси. Погана сторона полягає в тому, що складність вимагає від нас все більших зусиль, щоб бути в курсі останніх тенденцій веб-індустрії. Хороша ж сторона полягає в постійному зростанні інтернет-ринку, що в свою чергу дає нам, веб-фахівцям, більше клієнтів. Але не тільки це, в майбутньому ми побачимо все менше і менше поганих дизайнів, поганого коду і поганих проектів. Обмеження, накладені розвиненою індустрією і новітніми веб технологіями, не дозволять створити неякісний сайт або інтернет-ресурс, залишивши за бортом тих, хто не встиг за ними в своєму розвитку.

CRM допомагає *заробляти більше і легше*, а значить, з нею компанії зможуть успішніше розвивати свій бізнес. У 2015 році консалтингова компанія Capterra опитала 500 компаній і з'ясувала, що після впровадження CRM їх прибуток виріс від 25 до 35%.

У сучасному світі в бізнесі необхідність автоматизації різних процесів стала вже звичним явищем. Стає складно уявити собі складський або бухгалтерський облік без застосування спеціалізованого програмного забезпечення, торгові представники використовують спеціальні додатки для оформлення та відправки замовлення в офіс прямо з планшета або мобільного телефону, досить велика частина замовлень приходить з сайту вже у вигляді готових до обробки документів. Але при цьому

взаємовідносини з клієнтами, насамперед, в середньому і малому бізнесі, чомусь дуже часто ведуться без впровадження автоматизації і достатньої уваги до обліку.

Давайте уявимо ситуацію, коли в організації менеджери ведуть облік кожен по-своєму: хтось в електронних таблицях EXCEL, хтось від руки на папері, а хтось взагалі не вважає за потрібне фіксувати свій робочий процес. У підсумку, відстежити хто скільки напрацював, які замовлення були проведені, які дзвінки і кому відбувалися стає неможливим. Все це позначається на роботі фірми, вона втрачає клієнтів, гроші і час.

Вихід з цієї ситуації - автоматизація та стандартизація управління відносин з клієнтами, тобто впровадження CRM-системи.

2 СПЕЦІАЛЬНИЙ РОЗДІЛ

2.1 Тестове завдання для розробки CRM-системи

Отже, CRM система потрібна, щоб:

1. Не втратити потенційного клієнта, не пропустити жодного вхідного дзвінка і запиту. У бізнесі, особливо в малому та середньому, конкуренція дуже висока. Тому компанії докладають серйозні зусилля для того, щоб привернути до себе потік клієнтів;

2. Контролювати роботу співробітників і стандартизувати роботу з клієнтами. Інформацію про всі вхідні та вихідні контактах буде перебувати в одному сховищі, звідки її можна в будь-який момент отримати.

3. Накопичувати статистичну базу. Завдяки використанню CRM-системи вся робоча інформація збирається в одній загальній базі в стандартизованому вигляді, тобто керівник може аналізувати роботу і планувати подальшу більш усвідомлено.

4. Давати готові рішення, від яких можна відштовхуватися в побудові власної системи роботи. Різні інструменти системи самі підказують, які кроки варто зробити в процесі оптимізації роботи з клієнтами. [11]

Зараз можна виділити наступні кращі CRM-системи:

Простий бізнес

«Простий бізнес» - просунута CRM, яка комплексно автоматизує управління бізнесом і має вбудований модуль наскрізної аналітики. CRM-система працює і на стаціонарних комп'ютерах, і на смартфонах в форматі мобільного офісу.

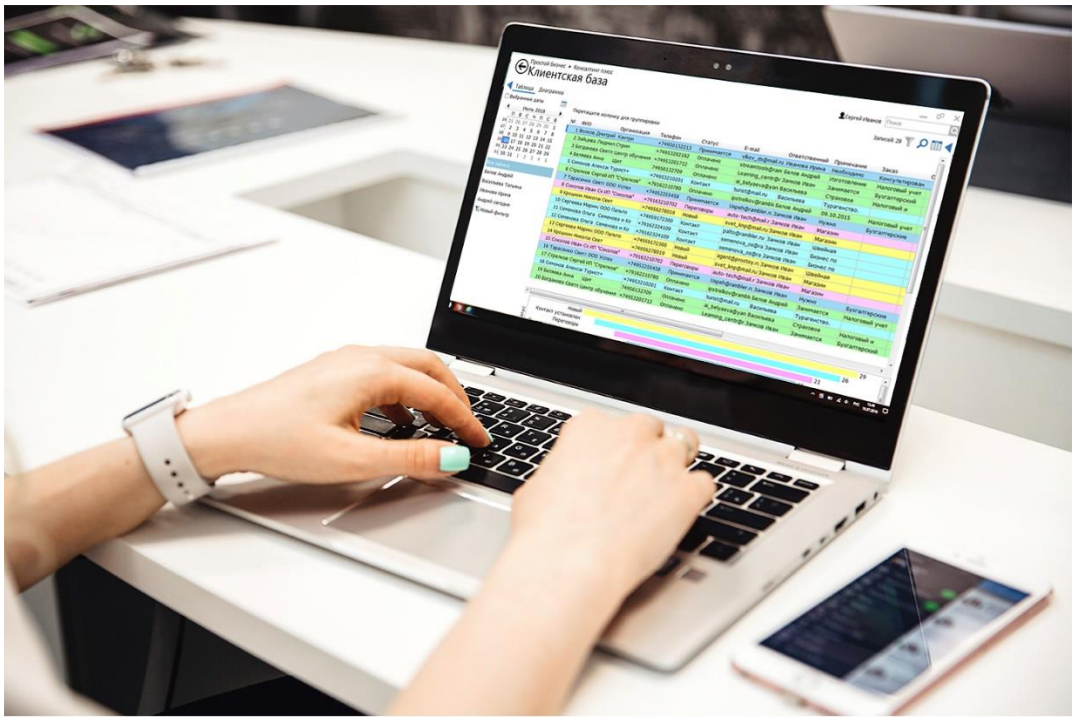

Рис. 2.1. «Простий бізнес»

Можливості програми:

- управління персоналом, проектами, завданнями і клієнтською базою;
- управління бухгалтерією і складом, документообігом, розсилками;
- продовження роботи при відключенні Інтернету;
- вбудовані комунікації;
- тестова версія - 30 днів повноцінного функціоналу безкоштовно;
- вбудований модуль наскрізної аналітики;
- безкоштовна базова версія для команд з п'яти співробітників.

Ціна:

профі – близько 27$/міс. за користувача;

VIP – 50$/міс. за користувача;

Коробка – 400$ одноразово за 30 співробітників.

Nethunt CRM

NetHunt CRM - проста і ефективна платформа, інтегрована в Gmail-пошту і G-Suite додатки. Система для управління взаємовідносинами з клієнтами впорядковує базу клієнтів, угоди і дозволяє відправляти масові

розсилки прямо з Gmail-пошти. Інтерфейс легко налаштовується під бізнес-процеси конкретної компанії.

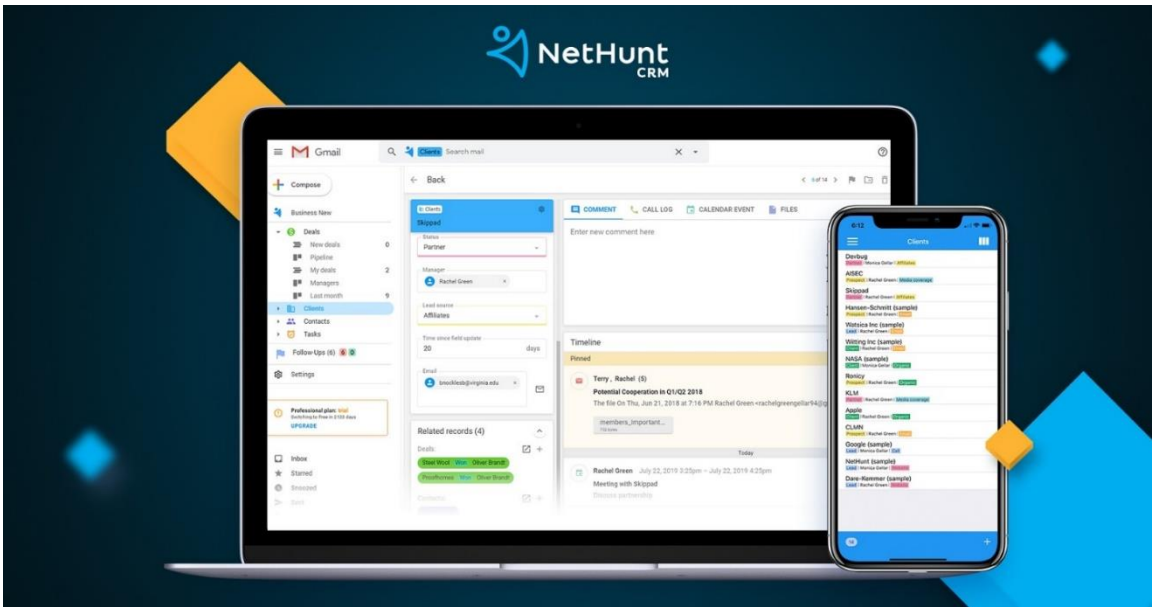

Рис. 2.2. Nethunt CRM

Можливості Nethunt CRM:

- Інтеграція з G Suite додатками;
- Детальна візуалізація воронки продажів;
- Гнучка настройка інтерфейсу за один день;
- Автоматичне оновлення даних про клієнта;
- Звіти за кількістю продажів і продуктивності співробітників;
- Масова email розсилка з відстеженням кліків і переглядів;
- Вся історія взаємодії з клієнтом доступна в один клік;
- Мобільний додаток.

Ціна: Professional - $10 за користувача в місяць, Professional Plus - $13 за користувача в місяць, Enterprise - $20 за користувача в місяць.

CRM Пачка

Пачка - мобільна хмарна CRM для малого і середнього бізнесу з інтуїтивним інтерфейсом і повноцінним додатком для смартфонів. Вона займає в рейтингу CRM-систем 2020 року 4 місце.

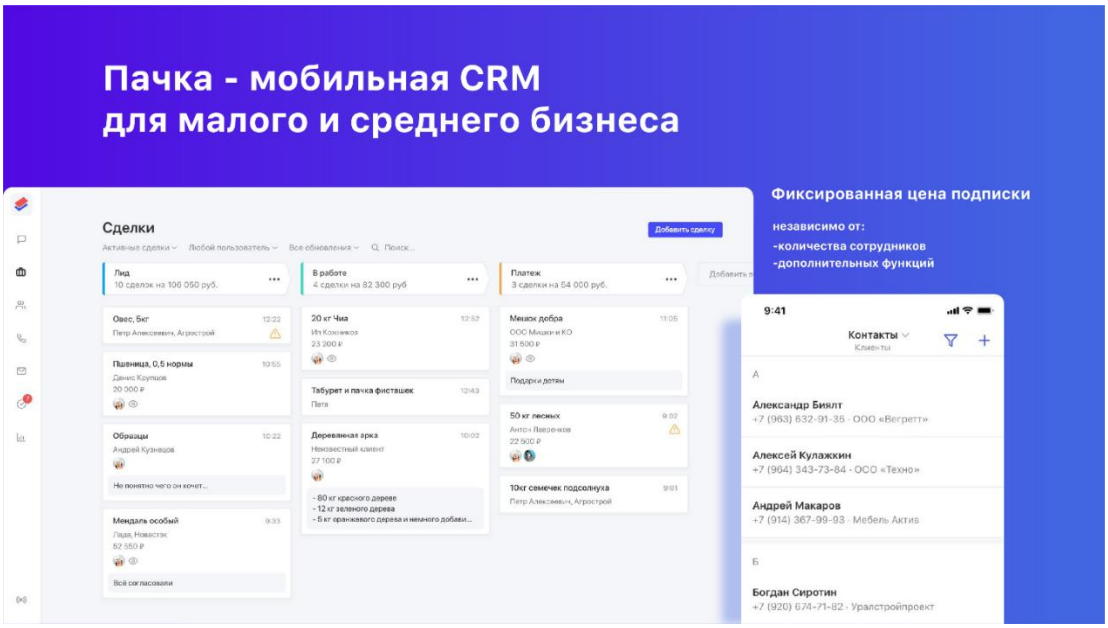

Рис. 2.3. CRM-пачка

Можливості CRM-системи Пачка:

- Робота з базою клієнтів
- Корпоративний месенджер
- Постановник задач
- Канбан-дошка з угодами
- Налаштування воронки продажів
- Імпорт бази клієнтів
- Складання звітів
- Веб-версія
- Мобільний додаток для Android і iOS
- Є нічна тема в мобільній версії
- Інтеграція з Tilda
- Інтеграція з Taplink
- Інтеграція з Yclients
- Інтеграція з електронною поштою
- Інтеграція з телефонією
- Швидкі відповіді в WhatsApp
- Інтеграція з рекламою Facebook і Instagram

- Тести та опитування
- Інтеграція з будь-яким сервісом за допомогою Rest API

Ціни:

Всі можливості за $27 на місяць незалежно від кількості співробітників і підключених міні-сервісів і інтеграцій.

amoCRM

amoCRM - одна з кращих CRM-систем для малого бізнесу. Головним чином тому, що інтерфейс дуже простий і зручний; є безліч опцій для кастомізації і інтеграції. Система доступна як мобільний додаток і їй можна користуватися навіть без установки. Користувач може заводити контакти, відстежувати операції, розподіляти зустрічі і виконувати інші завдання.

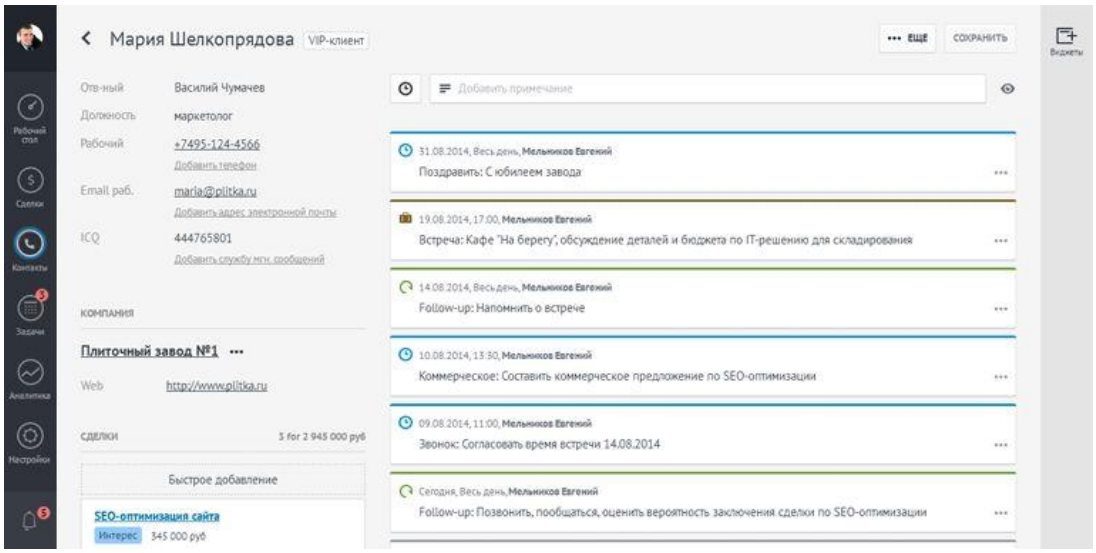

Рис. 2.4. amoCRM

У amoCRM спрощений інтерфейс, в порівнянні з аналогами, і схожі на адресну книгу. Система тісно інтегрована з веб-сайтом і GoogleAnalytics. Користувач створює форму, встановлює її на сайті, кожна залишена заявка фіксується як угода, до якої додається інформація з Analytics.

У amoCRM реалізовані наступні можливості:

- Угоди і контакти для управління продажами.
- Воронка продажів - для звітів.
- Завдання і нагадування.

- Аналіз продажів - моніторинг роботи співробітників і ефективності.
- Мобільні додатки - для Android і iOS.
- Інтеграція з сайтом - API, web-форми і лендінги.

Ціна: від $7 рублів на місяць (за користувача). [12]

Усі вищезгадані CRM-системи вимагають постійних фінансових вливань, так як для кожного нового користувача потрібно оплатити місячну підписку. Тому в наш час є актуальними приватні CRM-системи, створені під потреби конкретного клієнта.

Розглянемо реальний заказ клієнта. Клієнт хоче зручну, мінімалістичну crm-систему для технічного обслуговування клієнтів агро-сфери. В системі повинні бути:

1. Система авторизації, ролі користувачів з відповідними рівнями доступу:
   - адміністратор,
   - оператор,
   - тімлід.
2. Додавання, редагування, перегляд і видалення клієнтів в системі (картки клієнта з коментарем).
3. Мета для користувачів.
4. Картки завдань для користувачів.
5. Прогрес виконання завдань на головній сторінці.
6. Пошук по системі.
7. Фільтри для сторінок пошуку клієнтів, задач.

Клієнт хоче, щоб елементи системи були у сіро-зелених тонах, при цьому, однак перш за все, повинен бути мінімалістичний та зрозумілий кожному дизайн, щоб нові користувачі легко починали роботу в системі.

На етапі розробки дизайну системи був запропонований наступний стартовий дизайн сторінки (рис. 2.5):

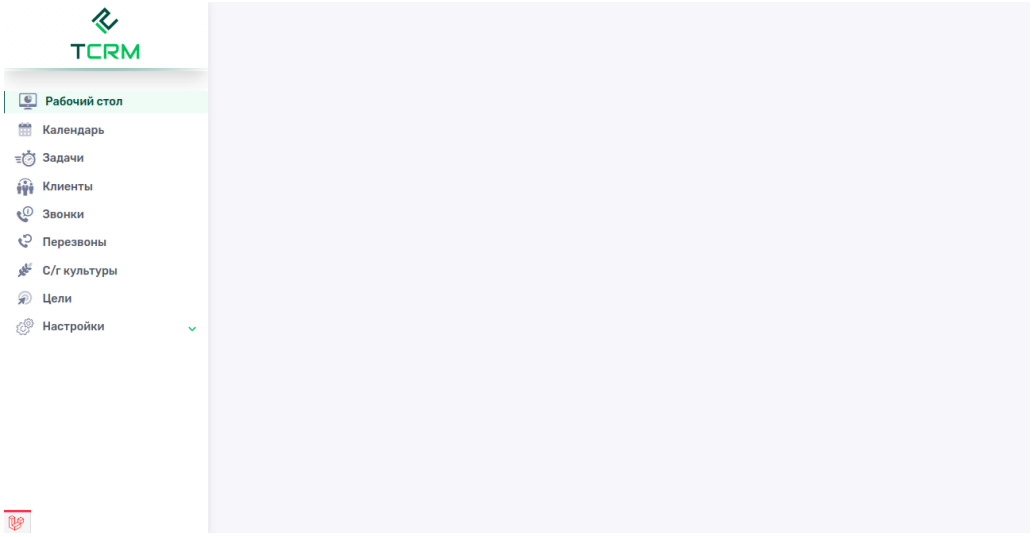

Рис. 2.5. Початкова пропозиція по дизайну агро-CRM-системи

Клієнт дав згоду, тому можна починати розробку.

2.2 Використання методу аналіза ієрархій для вибору CRM-системи

Метод аналізу ієрархій (MAI) — це структурований метод організації та аналізу складних рішень, заснований на математиці та психології. Він був розроблений Томасом Л. Сааті в 1970-х роках, який співпрацював з Ернестом Форманом для розробки вибору експертів у 1983 році, і з тих пір він широко вивчався та вдосконалювався. Він представляє точний підхід для кількісної оцінки ваги критеріїв прийняття рішень. Для оцінки відносної величини факторів за допомогою парних порівнянь використовується досвід окремих експертів.

Етапи методу:

1. Виділення проблеми. Визначення мети.
2. Виділення основних критеріїв та альтернатив.
3. Побудова ієрархії: дерево від мети через критерії до альтернатив (рис. 2.6).

4. Побудова матриці попарних порівнянь критеріїв за метою й альтернатив за критеріями.

5. Застосування методики аналізу отриманих матриць.

6. Визначення ваг альтернатив за системою ієрархії.

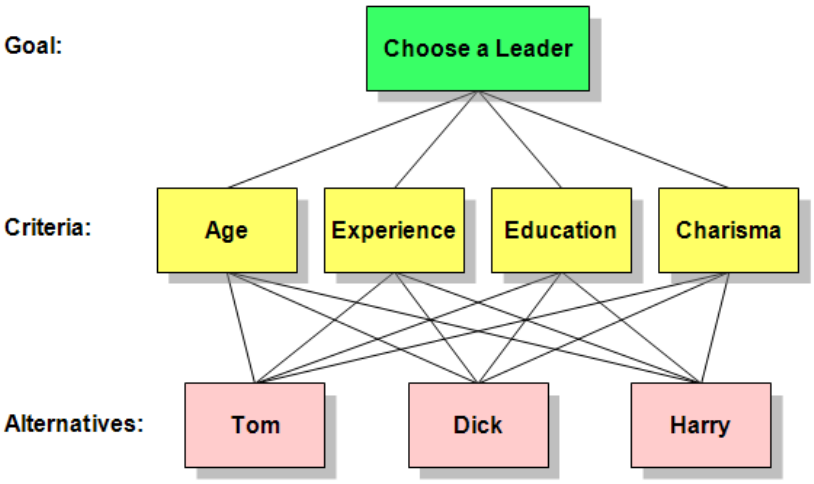

Рис. 2.6. Приклад дерева альтернатив

Розглянемо нашу проблему.

Мета: обрати вигідну CRM-систему для підприємства агро-бізнесу.

Основні критерії:

- Ціна
- Кількість користувачів у системі
- Відкритий вихідний код
- Чат всередині картки клієнта
- Можливість доопрацювання системи

Альтернативи (CRM-системи для вибору):

- Простий бізнес
- NetHunt
- amoCRM
- Приватна CRM-система

Порівняння критеріїв. Початково попарне порівняння за якісної шкалою,

з подальшим перетворенням в бали:

дорівнює, байдуже = 1

трохи краще (гірше) = 3 (1/3)

краще (гірше) = 5 (1/5)

значно краще (гірше) = 7 (1/7)

При проміжній думці використовуються проміжні бали 2, 3, 4, 5.

Відносна сила, величина або ймовірність кожного окремого об'єкта в ієрархії задається коефіцієнтом важливості відповідного елемента ієрархічного рівня, який визначається розрахунком відповідного йому елемента власного вектора матриці пріоритетів, нормалізованого до одиниці.

Порівняно тривала, навіть за умови застосування ЕОМ, процедура визначення власних векторів матриць піддається наближеному рішенню за допомогою обчислення геометричної середньої (витягується корінь *n*-го ступеня, де *n* - розмірність матриці порівнянь з добутку елементів кожного рядка). Для кожного елемента обчислюється середнє значення значущості за формулою:

$$F_i = \left[\prod_{j=1}^{n} f_{i,j} \right]^{1/n}.$$

При цьому діє принцип зворотної пропорційності: якщо при порівнянні одного фактора і з іншим j отримано a (j, i) = b, то при порівнянні другого фактора з першим a (j, i) = 1 / b.

Нехай А1, ..., Ап - безліч з *n* елементів, W1, ..., W "- співвідносяться так, як показано на рис. 2.7:

	A_1	A_2	...	A_n
A_1	1	W_1/W_2		W_1/W_n
A_2	W_2/W_1	1		
...	...			
A_n	W_n/W_1			1

Рис. 2.7. Матриця порівняння

При порівнянні елементів, що належать одному рівню ієрархії, ЛПР висловлює свою думку, використовуючи одне з визначень шкали інтенсивності. У матрицю порівняння заноситься відповідне число. При бажанні ОПР може використовувати і парні цілі числа, висловлюючи проміжні рівні переваги за важливістю.

Складаємо матрицю порівняння для критеріїв (табл. 2.1):

Таблиця 2.1

Матриця порівняння для усіх критеріїв

	цена	пользователи	код	чат	доработка
цена	1,00	2,00	3,00	0,50	0,33
пользователи	0,50	1,00	2,00	0,33	0,25
код	0,33	0,50	1,00	0,25	0,20
чат	2,00	3,00	4,00	1,00	0,50
доработка	3,00	4,00	5,00	2,00	1,00

Таблиця 2.2

Матриця порівняння за критерієм «ціна»

цена	пб	нет	амо	приватная
пб	1,00	7,00	3,00	5,00
нет	0,14	1,00	0,20	0,33
амо	0,33	5,00	1,00	3,00
приватная	0,20	3,00	0,33	1,00

Таблиця 2.3

Матриця порівняння за критерієм «кількість користувачів»

пользователи	пб	нет	амо	приватная
пб	1,00	3,00	0,33	5,00
нет	0,33	1,00	0,20	3,00
амо	3,00	5,00	1,00	7,00
приватная	0,20	0,33	0,14	1,00

Таблиця 2.4

Матриця порівняння за критерієм «вихідний код»

код	пб	нет	амо	приватная
пб	1,00	0,33	0,20	0,14
нет	3,00	1,00	0,33	0,20
амо	5,00	3,00	1,00	0,33

приватная	7,00	5,00	3,00	1,00

Таблиця 2.5.

Матриця порівняння за критерієм «чат у картці клієнта»

чат	пб	нет	амо	приватная
пб	1,00	0,20	0,33	0,14
нет	5,00	1,00	3,00	0,33
амо	3,00	0,33	1,00	0,20
приватная	7,00	3,00	5,00	1,00

Таблиця 2.6.

Матриця порівняння за критерієм «можливість доопрацювання»

доработка	пб	нет	амо	приватная
пб	1,00	0,33	0,20	0,14
нет	3,00	1,00	0,33	0,20
амо	5,00	3,00	1,00	0,33
приватная	7,00	5,00	3,00	1,00

Наступний етап – нормування матриць. Знаходяться суми елементів кожного стовбця, усі елементи матриці діляться на суму елементів відповідного стовбця:

Таблиця 2.7.

Обчислення сум елементів кожного стовбця

	цена	пользователи	код	чат	доработка
цена	1,00	2,00	3,00	0,50	0,33
пользователи	0,50	1,00	2,00	0,33	0,25
код	0,33	0,50	1,00	0,25	0,20
чат	2,00	3,00	4,00	1,00	0,50
доработка	3,00	4,00	5,00	2,00	1,00
сумм	6,83	10,50	15,00	4,08	2,28

Таблиця 2.8.

Знаходження середнього значення кожного критерію

	цена	пользователи	код	чат	доработка	срзнач
цена	0,15	0,19	0,20	0,12	0,15	0,16
пользователи	0,07	0,10	0,13	0,08	0,11	0,10
код	0,05	0,05	0,07	0,06	0,09	0,06
чат	0,29	0,29	0,27	0,24	0,22	0,26
доработка	0,44	0,38	0,33	0,49	0,44	0,42
сумм	1,00	1,00	1,00	1,00	1,00	1,00

Отриманий стовбець задає «ваги» критеріїв з точки зору поставленої мети. Цей стовбець називають «ваговим стовбцем критеріїв» (табл. 2.8):

Таблиця 2.9.

Ваговий стовбець критеріїв

Критерій	Вага	Вага в відсотках
цена	0,16	16,11%
пользователи	0,10	9,86%
код	0,06	6,24%
чат	0,26	26,18%
доработка	0,42	41,62%

З точки зору задоволення нашої мети найбільш вагомою є можливість доопрацювання системи (41,62%), далі йде чат в картці клієнта (26,18%), потім йде ціна (16,11%). Далі найменш значущі критерії – кількість користувачів (9,86%) та вихідний код (6,24%). Повторюємо нормування для кожного з критеріїв та отримаємо стовбці (вектори) вагових коефіцієнтів об’єктів порівняння з точки зору відповідності окремим критеріям (Додаток Г).

Отриманий результат (табл. 2.10):

Таблиця 2.10

Ваги альтернатив

Альтернатива	Вага	Вага у відсотках
пб	0,16	16%
нет	0,15	15%
амо	0,26	26%
приватная	0,44	44%
сумм	1,00	

Таким чином, альтернатива «Приватна CRM-система» є найбільш привабливою для досягнення нашої мети.

2.3 Створення приватної CRM-системи

Для створення CRM-системи був обраний php-фреймворк Laravel7. Вибір зумовлений високою функціональністю фреймворка та простотою у використанні. Laravel-проект встановлюється з-під коробки Composer.

Основи роботи з laravel:

1. Міграція баз даних
2. MVC
3. «Роутінг»

Словосполучення «Міграція баз даних» звучить швидше за все кілька лякаюче для новачка. Для роботи з міграціями достатньо знати, для чого потрібна база даних (в принципі, очевидно що для зберігання інформації) і розуміння чогось більше ніж «SELECT * FROM Customers» в SQL-командах. Отже, міграція - це щось на зразок системи контролю для перенесення ваших таблиць в БЗ с допомогою конструктора таблиць. Міграція дозволить вам уникнути помилок і конфліктів під час конструювання таблиць в базі даних для великого проекту разом з учасниками іншої команди. Крім того, це дозволить взаємодіяти з базою даних не за допомогою таких інструментів, як MySQL WorkBench або PhpMyAdmin, а безпосередньо з коду, в залежності від потреб вашого проекту в таблицях даних. Додавання і видалення таблиць записується в історії міграцій, і тепер хоча б у тімліда стане на одну головну біль менше.

ORM - система об'єктно-реляційного відображення, яка пов'язує бази даних з концепціями об'єктно-орієнтованого програмування. Звучить на перший погляд страшно, проте це безпосередньо пов'язано з міграцією баз даних: на кожну таблицю створюється свій клас - **модель**, який використовується тільки для роботи з цією таблицею. Це дозволяє не розводити зайвої роботи в самій базі даних, а взаємодіяти з нею знову ж безпосередньо з проекту. У підсумку це виходить і зручніше, і надійніше.

Звичайно, на освоєння команд і особливостей генерації моделей піде якийсь час, але на створення великого проекту з величезною кількістю таблиць його піде куди більше. [13]

У фреймворку Laravel все працює за принципом MVC (англ. Model View Controller - модель-уявлення-контролер). Припустимо, у нас є інтернет-магазин побутової техніки. Тут моделями будуть: товар, замовлення, постачальник, користувач.

Controller - логіка, логіка і тільки логіка роботи того чи іншого елемента (часто пов'язується з моделлю).

Blade - шаблони дають можливість зручно зв'язувати ваші уявлення виду. Це і є фронт-енд нашого проекту або його уявлення (view). Ось як би це виглядало на нативному PHP:

```
<?php  include($_SERVER['DOCUMENT_ROOT']."/shop/includes/header.inc.php");  ?>
<div class="catalog">
    <div class="catalog-filters"> </div>
    <div class="catalog-products"> </div>
    <div class="catalog-pagination"> </div>
</div>
<?php  include($_SERVER['DOCUMENT_ROOT']."/shop/includes/footer.inc.php");  ?>
```

А ось так виглядає шаблон blade:

```
<html>
  <head>
  </head>
  <body>
      @extends(‘header')
      @extends(‘catalog')
      @extends(‘footer’)
  </body>
</html>
```

Погодьтеся, виглядає значно акуратніше! Звичайно, насправді за цим стоїть код контролера і експорт з інших Blade-шаблонів, однак на великому проекті це буде тільки зручніше.

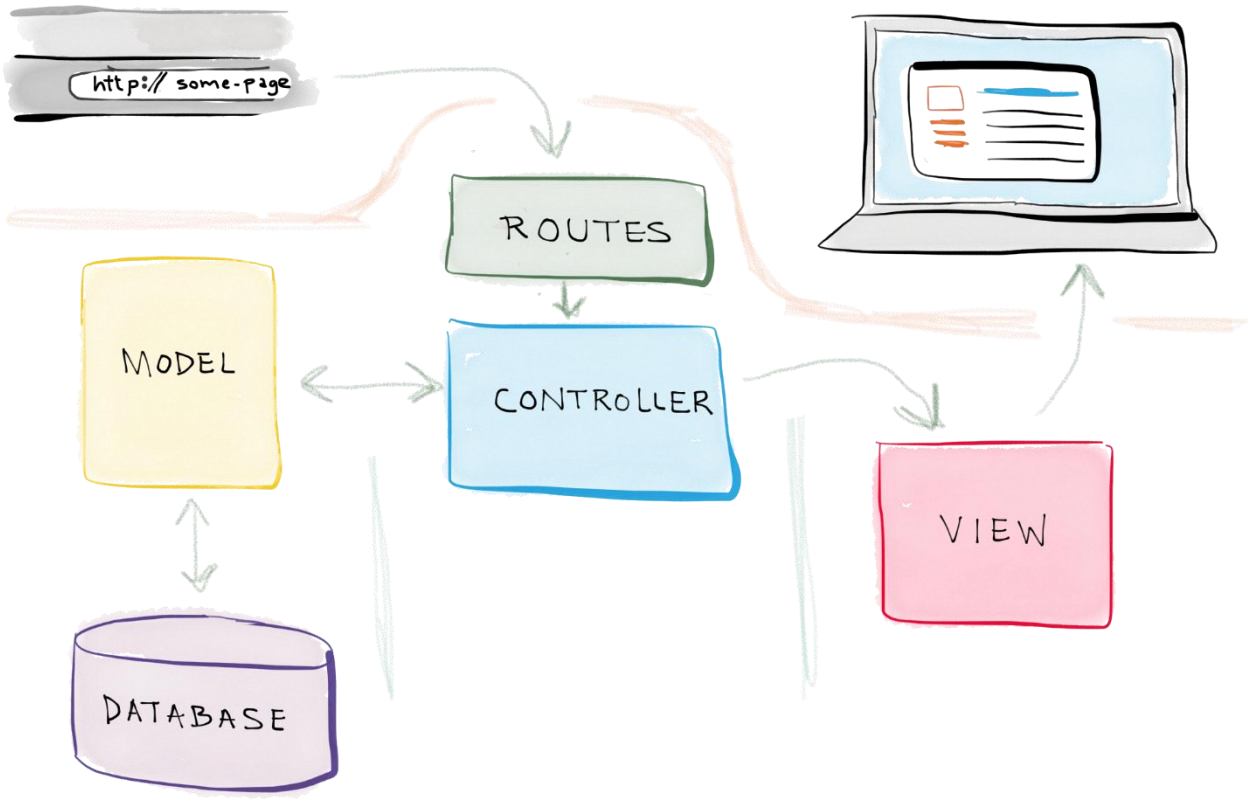

Рис. 2.8. Схема MVC-архітектури Laravel-проекту

1. Отримання запита користувача з браузера
2. Вибір контролера відповідно до маршруту користувача
3. Взаємодія з моделлю даних
4. Контролер викликає відповідний файл view
5. Візуалізація поданих контролером даних у браузері користувача

Laravel практично не вимагає додаткової настройки з коробки. Ми можемо почати розробку.

Цей фреймворк пропонує прекрасну документацію по розробці, напевно, кращу, з тих що я бачив. Весь функціонал описаний чітко і лаконічно за посиланням https://laravel.com/docs/7.x (до речі, в момент написання даної роботи вийшло оновлення Laravel 8, але я продовжив працювати зі звичною мені версією).

Важливо сказати, що розробка проекту ведеться на операційній системі Linux Ubuntu 20.04, так як Ubuntu надає безкоштовний вбудований віртуальний сервер і ми зможемо «поставити на ноги» проект локально на комп'ютері, тим самим оцінивши, як він буде виглядати в мережі інтернет.

Ввівши одну команду:

```
composer global require laravel/installer
```

ми встановимо laravel на наш комп'ютер. Далі, для створення проекту ми використовуємо команду:

```
laravel new crm-system
```

і установник laravel встановить всі потрібні файли для початку роботи в вихідній папці.

Початковий проект має наступний вигляд:

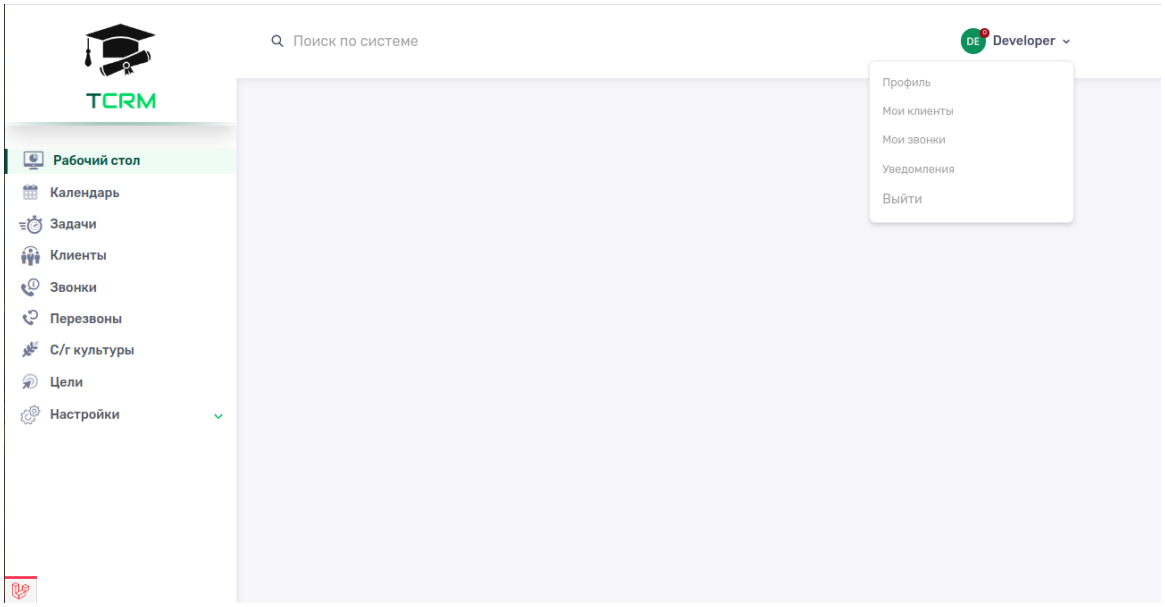

Рис. 2.9. Вигляд початкового проекту

Для розробки фронтенду проекта використовувався css-фреймворк TailwindCSS. Його великою перевагою над іншими фрейморками є те, що стилі для розмітки сторінки створені заздалегідь і мають схожий з назвою стилю клас. Наприклад, для того щоб пофарбувати текст в синій колір, можна було б використовувати

```
1.  <p class="blue"> Синій </p>
```

в html, але також додати стиль для класу `blue` в css:

```
1.  p .blue {
2.      color: #0000ff;
```

```
3. }
```

TailwindCSS має десятки тисяч прописаних стилів з відповідними класами. Вам лише потрібно підключити вихідні файли і почати роботу з ними. Для прикладу вище буде досить додати клас `text-blue-900:`

```
2. <p class="text-blue-900"> Синій </p>
```

Даний клас прописаний у вихідних файлах tailwindcss, нам залишається лише користуватися ними. Це значно спростить розробку. Щоб притримуватися одного стилю, ми будемо повторно використовувати класи tailwindcss:

```
text-gray-400, text-2xl, font-bold, w-full, flex, flex-col, cursor-pointer, etc.
```

Їх визначення виходить з назви, наприклад:

Font-bold - текст жирним, w-full - ширина 100%, flex - display: flex і т.п. В першу чергу, потрібно визначити моделі нашої системи. Це:

- Користувач – user,
- Задача – task,
- Мета – goal,
- Клієнт – customer,
- Культура – crop,
- Коментар – comment,
- Дзвінок – call.

Для кожного елемента меню будуть створені роути (те, що знаходиться після domain.com/...), наприклад для сторінки задач це буде *domain.com/**tasks***. Їх можна вибирати довільно і прописувати в файлах *web.php, api.php*. Кожному роуту повинен відповідати контролер або метод контролера. Стандартні методи контролерів: *index* - загальний вигляд всіх елементів (наприклад, всі клієнти), *show* - сторінка одного елемента, *create* - сторінка створення елемента, *store* - функція створення елемента, *update* - функція редагування елемента, *edit* - сторінка редагування елемента *destroy* - функція видалення елемента.

Замовник уточнив, що досить буде зробити CRUD с/г культур.

```php
<?php

namespace App\Http\Controllers\Crops;

use App\Models\Crops\Crop;
use App\Http\Requests\Crops\Create;
use App\Http\Requests\Crops\Update;
use Illuminate\Auth\Access\AuthorizationException;
use Illuminate\Contracts\Routing\ResponseFactory;
use Illuminate\Http\Request;
use Illuminate\Http\Response;
use App\Http\Controllers\Controller;

class CropsController extends Controller
{
    public function index()
    {
        return Crop::query()->paginate();
    }

    public function store(Create $request)
    {
        return response(Crop::create($request->validated()), 201);
    }

    public function show(Request $request, Crop $crop)
    {
        return response($crop);
    }

    public function update(Update $request, Crop $crop)
    {
        return response()->json(tap($crop)->update($request->validated()), 202);
    }

    public function destroy(Crop $crop)
    {
        $this->authorize('delete', $crop);

        $crop->delete();

        return response('Deleted', 204);
    }
}
```

Розглянемо на прикладі CropsController - контролер логіки роботи з сільськогосподарськими культурами або backend crud-а культур:

Функція `index()` - виведення списку культур. У цій функції описана логіка виведення: вивести всі моделі Crop і спагінувати їх (зробити пагінацію сторінок).

Функція `store()` - функція створення культури. Аргументом функції є запит до сервера на створення культури в БД, функція повертає відповідь сервера, чи була створена культура з очікуванням відповіді 201.

Функція `show()` - вивід сторінки культури, яка в аргументі функції - виводить дану культуру.

Функція `update()` - функція редагування культури. В аргументах функції є сама культура, яку потрібно відредагувати, вона редагується в БД і відповідь сервера, чи була культура відредагована (редагування, як і створення виробляється в файлах html - тобто фронтенд проекту).

Функція `destroy()` - функція видалення культури. Аргументом є обрана культура, перевіряється можливість видалення (в зв'язку з роллю користувача), культура видаляється з БД, повертається відповідь сервера.

CRUD можна реалізувати і на одній сторінці по роуту **/crops**, але laravel надає нам можливість зареєструвати окремі динамічні посилання для кожної сторінки. Це прописується в файлах *web.php, api.php*:

```
1.  Route::apiResource('crops', 'Crops\CropsController');
```

Тепер можна використовувати роути **crops, /crops/1, /crops/30** і т.д. використовуючи різні методи (рис. 2.10):

```
anton@anton-Latitude-E5440:~/projects/crm$ php artisan route:list --name=crops
+--------+-----------+-----------------+------------------+-------------------------------------------------------+--------------+
| Domain | Method    | URI             | Name             | Action                                                | Middleware   |
+--------+-----------+-----------------+------------------+-------------------------------------------------------+--------------+
|        | GET|HEAD  | api/crops       | api.crops.index  | App\Http\Controllers\Crops\CropsController@index      | api,auth:api |
|        | POST      | api/crops       | api.crops.store  | App\Http\Controllers\Crops\CropsController@store      | api,auth:api |
|        | GET|HEAD  | api/crops/{crop} | api.crops.show   | App\Http\Controllers\Crops\CropsController@show       | api,auth:api |
|        | PUT|PATCH | api/crops/{crop} | api.crops.update | App\Http\Controllers\Crops\CropsController@update     | api,auth:api |
|        | DELETE    | api/crops/{crop} | api.crops.destroy | App\Http\Controllers\Crops\CropsController@destroy   | api,auth:api |
+--------+-----------+-----------------+------------------+-------------------------------------------------------+--------------+
```

Рис. 2.10. Роутинг для сторінок с/г культур

Реалізуємо сторінку списку культур, сторінку створення культури, редагування існуючої і додамо функції редагування і видалення на головній сторінці за роутом **/crops**:

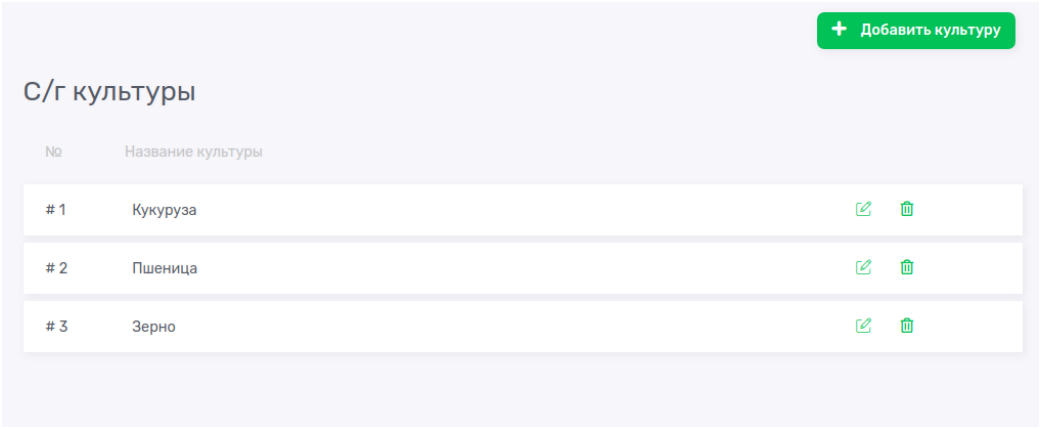

Рис. 2.11. Сторінка списку с/г культур (**/crops**)

Тут ми можемо побачити кнопки для додавання культури, редагування і видалення культур. Все це працює синхронно з базою даних, в реальному часі.

Нам потрібно виконати подібну роботу з наступними компонентами: клієнти, завдання, мета, дзвінки, створити віджети виконаної роботи на головній сторінці. Для клієнтів будуть характерні наступні поля (рис. 2.12 - 2.13):

1. Компанія
2. Ім’я клієнта
3. Телефон
4. Додатковий телефон
5. Email
6. Область, район, місто, село
7. Відповідальний (зв’язок в БД з users)
8. Статус клієнта
9. Об’єм землі

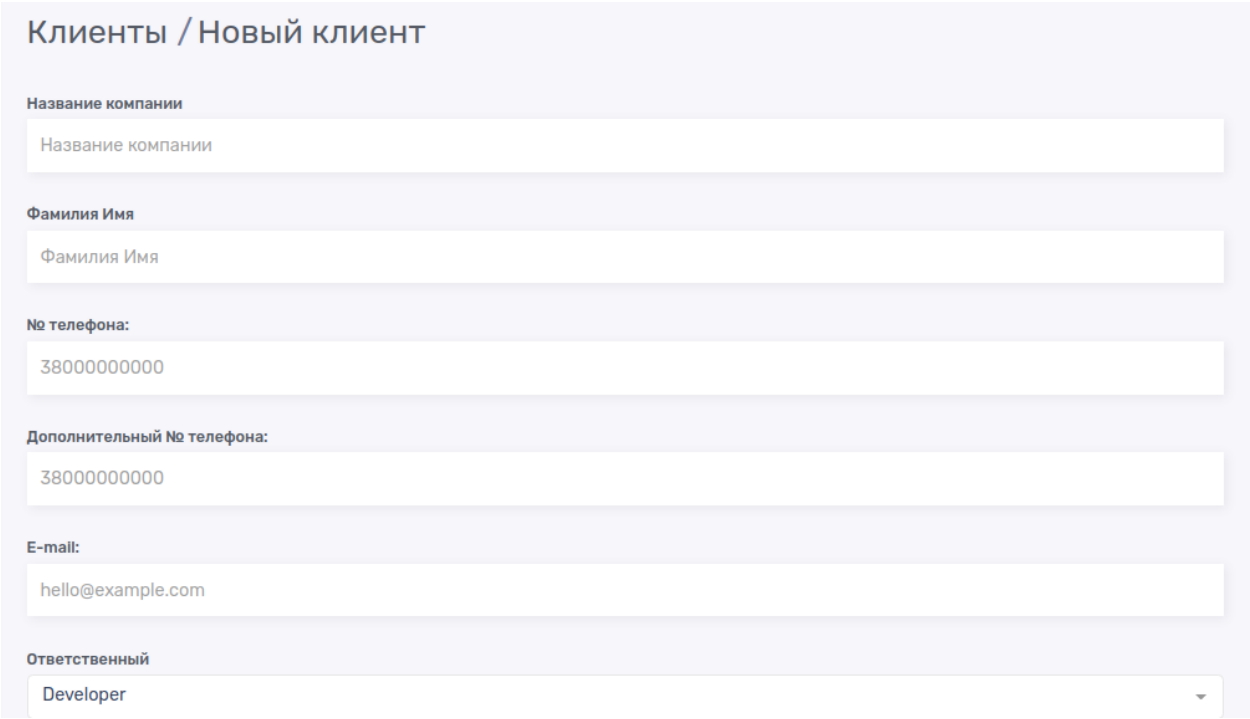

Рис. 2.12. Створення нової карточки клієнта (**/crops/create**)

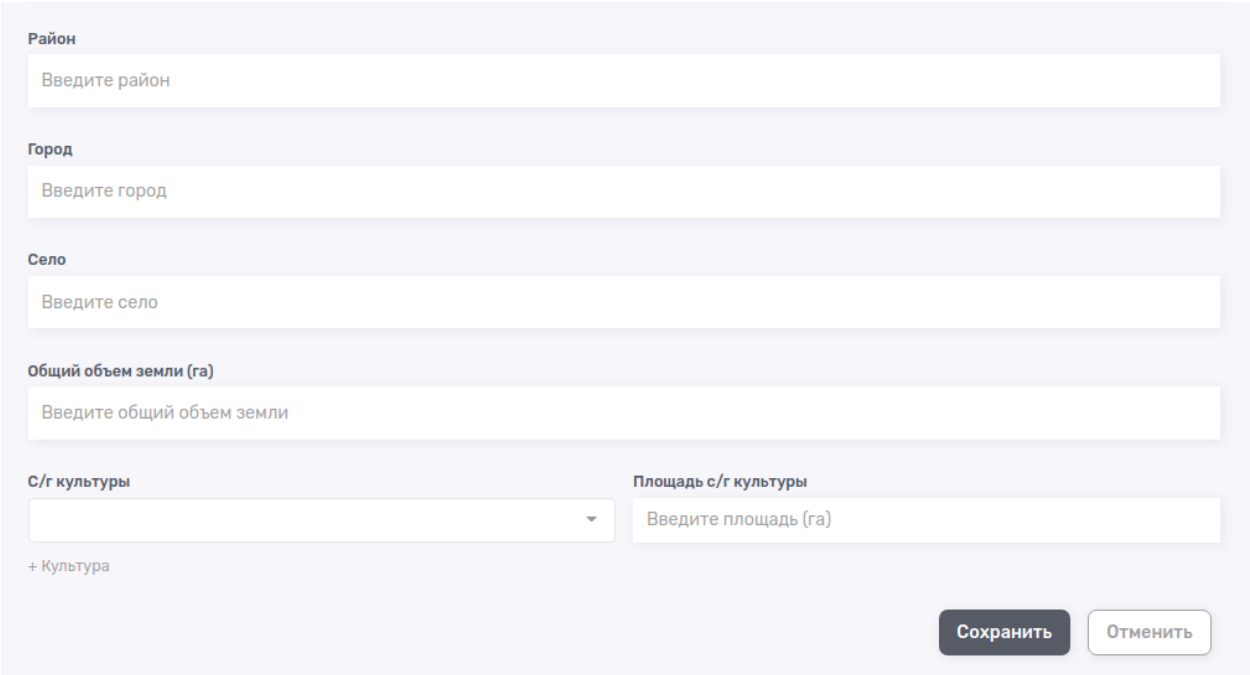

Рис. 2.13. Продовження створення нової карточки клієнта

Поля «ім'я» і «номер телефону» є обов'язковими, так як без них неможливий контакт з клієнтом. Всі інші поля мають властивість *nullable()* - тобто може бути порожнім. Сторінка **/crops** буде виглядати наступним чином (рис. 2.14):

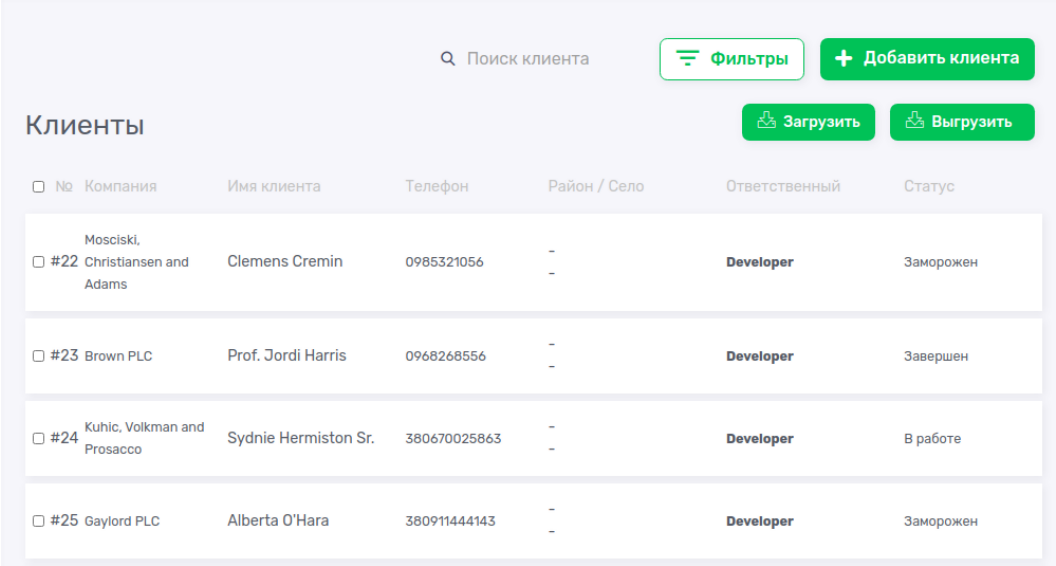

Рис. 2.14. Сторінка списку клієнтів (**/customers**)

Продовжимо роботу за алгоритмом: логіка в контролері, створення роута, оформлення фронтенду. Таким чином, були розроблені наступні сторінки (рис. 2.15 - 2.19):

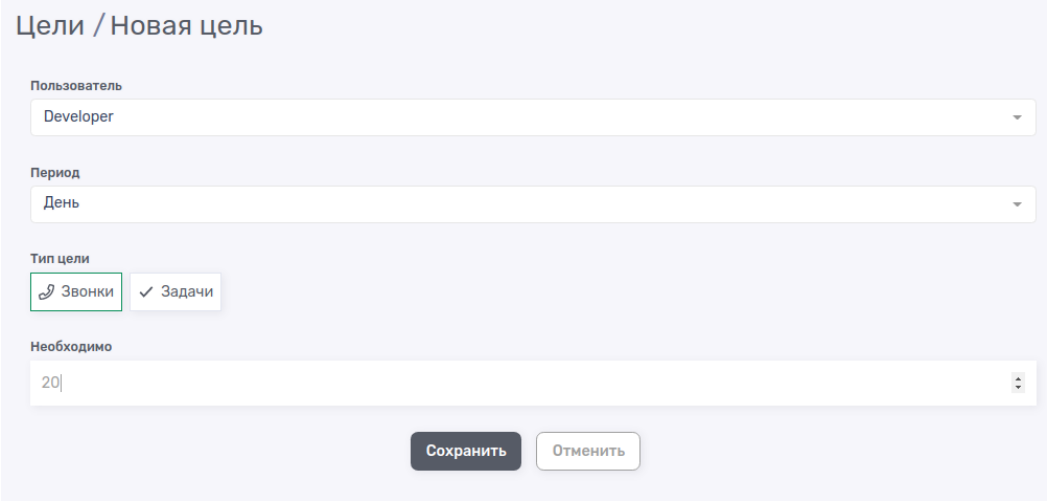

Рис. 2.15. Створення нової мети для користувача (**/goals/create**)

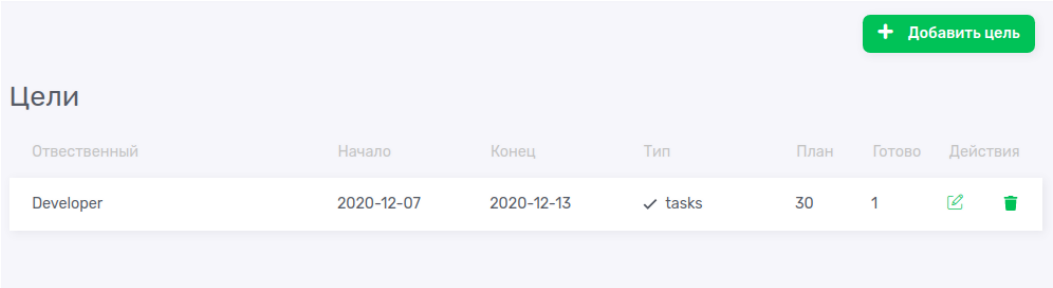

Рис. 2.16. Сторінка списку цілей для користувача (**/goals**)

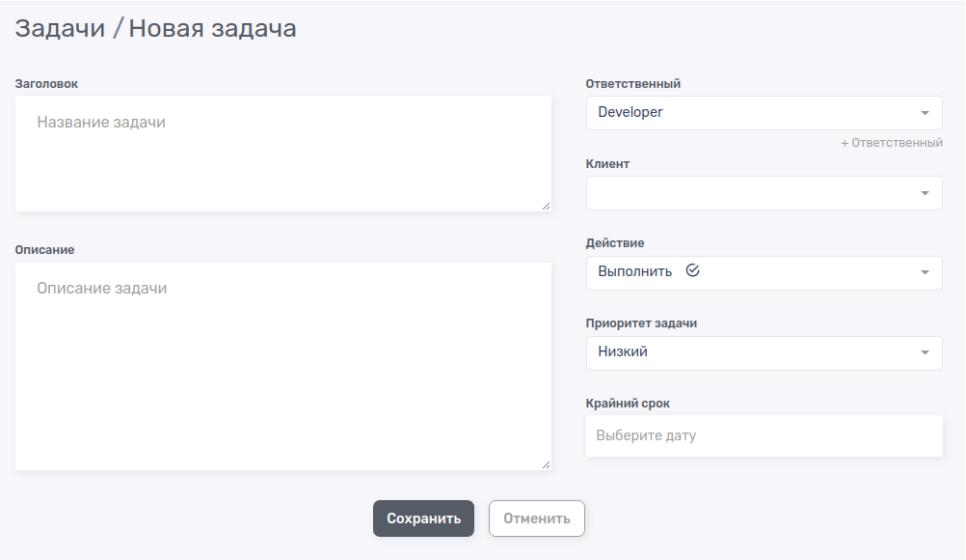

Рис. 2.17. Створення нової задачі для користувача (**/tasks/create**)

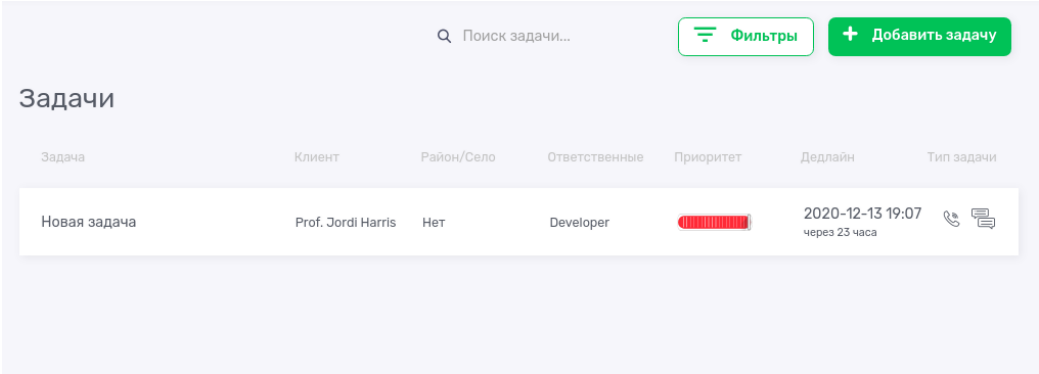

Рис. 2.18. Сторінка списку задач для користувача (**/tasks**)

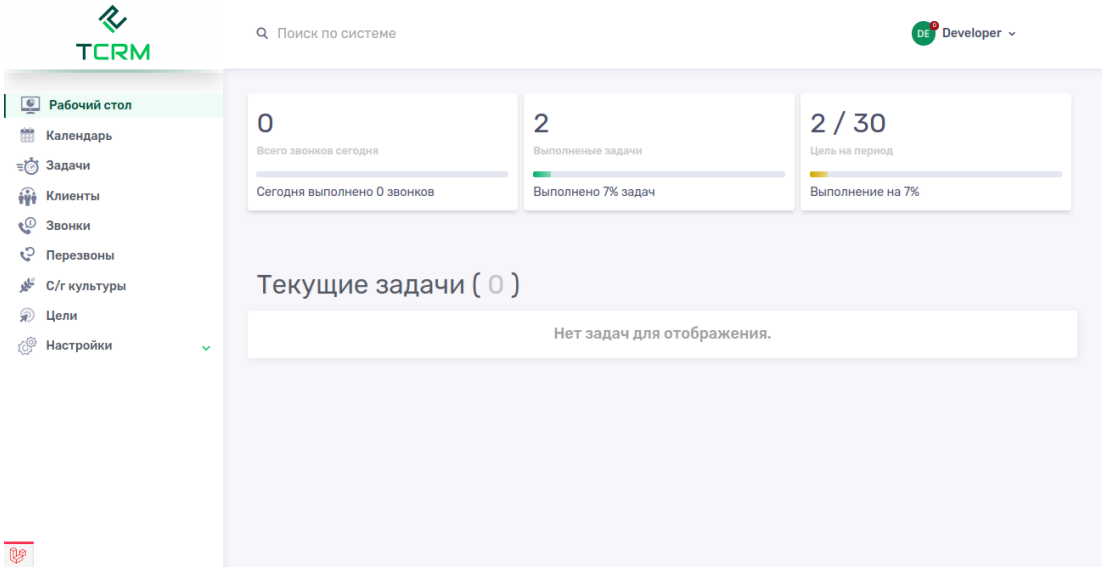

Рис. 2.19. Робоче середовище користувача (/ створює редірект на **/dashboard**)

2.4 Висновки до розділу

Найбільшу користь CRM-системи приносять компаніям сектора «business-to-business», що використовують метод прямих продажів товарів і послуг кінцевому споживачу. Продукт або послуга тут - товар, вигоду від використання якого споживачеві не завжди видно відразу. Або ж вони (товар або послуга) знаходяться в високо-конкурентному ринку, що надає клієнту різноманіття вибору. Прикладами таких компаній є комп'ютерні, рекламні, консалтингові фірми, банки та ін. Прямі продажі мають на увазі безпосереднє тривалу взаємодію співробітників компанії-продавця з клієнтом. І успіх взаємодії часто залежить від того, наскільки якісно менеджер підготувався до зустрічі з клієнтом. Саме якість роботи з клієнтом, що забезпечується повнотою інформації про нього (клієнта), стає найважливішою конкурентною перевагою компанії. Особливо у випадках, коли бізнес компанії побудований на угодах з тривалим циклом їх здійснення. Можливість відстежити історію роботи з клієнтом, спрогнозувати його реакцію на їхні дії і т. п. - все це різко збільшує шанси компанії на успішне завершення угоди.

ВИСНОВКИ

В цілому впровадження CRM в умови сучасного ринку дозволить реалізувати такі можливості:

- взяти контроль над усіма каналами комунікації з клієнтами;
- збільшити швидкість і якість обробки вхідної черги і як наслідок збільшити зростання продажів;
- створити клієнтську базу, налаштовану під конкретний вид бізнесу;
- налагодити контроль і прозорість ведення угод співробітниками;
- автоматизувати документообіг в компанії, виключити помилки у формуванні документів;
- створити єдине комунікаційне простір для співробітників;
- автоматизувати бізнес-процеси в компанії;
- аналізувати якість ведення угод за допомогою інструменту воронка продажів, бачити на яких етапах угоди провалюються, працювати над слабкими сторонами і удосконалювати сильні;
- аналізувати ефективність роботи співробітників. [14]

З мінусів:

- вартість, впровадження CRM - системи вимагає грошових коштів;
- необхідність перебудовувати схему роботи компанії.

Впроваджувати CRM систему або не впроваджуються звичайно ж кожна компанія повинна самостійно, зваживши всі за і проти. Ринок CRM-систем зараз активно розвивається і все більше компаній використовують як концепцію CRM, так і CRM-системи у веденні бізнесу і компанії, які відкладають впровадження ризикують опинитися в рядах «наздоганяючих».

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Николаевская, Т. Н. АКТУАЛЬНІСТЬ ТА ПРОБЛЕМИ ВПРОВАДЖЕННЯ В ПРАКТИКУ РОСІЙСЬКИХ ПІДПРИЄМСТВ CRM-СИСТЕМ // Молодь і наука: Збірник матеріалів VI Всеросійської науково-технічної конференції студентів, аспірантів і молодих вчених [Електронний ресурс]. - Красноярськ: Сибірський федеральний ун-т, 2011.

2. Вилегжаніна А.О. CRM-системи: навчальний посібник / Москва, Берлін: Директ-Медіа, 2016. - 99 с.

3. Стандартизація бізнес процесів компанії - призначення та застосування: під ред. В.В. Льва: веб-сайт. URL: https://bank-explorer.ru/optimizaciyaprocessov/standartizaciya-processov.html (дата звернення: 01.10.2020).

4. Офіційний сайт «Портал знань»: веб-сайт. URL: http://www.znannya.org/ (дата звернення: 03.10.2020)

5. Петлюшкин А.В., HTML в Web-дизайне. – СПб.: БВХ-Петербург, 2004. – 400 с.

6. Основи CSS – Вчимо веб-розробку | MDN: веб-сайт. URL: https://developer.mozilla.org/uk/docs/Learn/Getting_started_with_the_web/CSS_basics (дата звернення: 03.10.2020).

7. 7 причин, за якими PHP хороший для розробки бізнес-проектів: веб-сайт. URL: https://stfalcon.com/ru/blog/post/PHP-advantages-for-business (дата звернення: 06.10.2020).

8. 8 кращих PHP Framework для веб-розробників: веб-сайт. URL: https://www.hostinger.ru/rukovodstva/8-luchshih-php-framework-dla-web-razrabotchikov/ (дата звернення: 07.10.2020).

9. Front end та back end – Вікіпедія: веб-сайт. URL: https://ru.wikipedia.org/wiki/%D0%A4%D1%80%D0%BE%D0%BD%D1%82%

D0%B5%D0%BD%D0%B4_%D0%B8_%D0%B1%D1%8D%D0%BA%D0%B5%D0%BD%D0%B4 (дата звернення: 09.10.2020).

10. CRM система для управління взаємовідносинами з клієнтами | Terrasoft: веб-сайт. URL: https://www.terrasoft.ua/page/crm-definition (дата звернення: 11.10.2020).

11. Технологии повышения спроса на туристические услуги / Рассуль Салим Масир Аль Малеки // Транспортное дело России. 2015. № 5. С. 46-47

12. Топ-10 кращих CRM-систем для бізнесу 2020. Рейтинг та порівняння: веб-сайт. URL: https://www.kickidler.com/ru/for-it/methods-of-working/top-10-luchshix-crm-sistem.-rejting-i-sravnenie.html (дата звернення: 20.10.2020).

13. Laravel: пояснюємо основні поняття. Частина перша: «Теорія»: веб-сайт. URL: https://habr.com/ru/company/otus/blog/470794/ (дата звернення: 24.10.2020).

14. Лещёв В. А. Эффективность применения CRM-системы // Молодой ученый. — 2016. — №12. — С. 165-168.

15. Ус С.А. Теорія прийняття рішень. Методичні рекомендації до виконання курсової роботи студентами напряму підготовки 6.040303 Системний аналіз, 2015 рік.

ДОДАТКИ

Додаток А

Відомість матеріалів кваліфікаційної роботи

№ з/п	Позначення	Назва	Кількість	Примітки
1				
2		Документація		
3				
4	САіУ.РД.20.05.ПЗ	Пояснювальна записка	54	Формат А4
5				
6	САіУ.РД.20.05.ДМ	Демонстраційні матеріали	10	Презентація на CD-R
7				
8	САіУ.РД.20.05.КР	Копія роботи	1	Диск CD-R
9				
10				
11				
12				
13				
14				
15				
16				
17				
18				

					САУ.КР.21.02.ДА.ПЗ.			
Змін.	Аркуш	№ докум.	Підпис	Дата				
Розроб.		Василенко Ю.А.			Матеріали кваліфікаційної роботи	Літ.	Аркуш	Аркушів
К. Розд.		Слєсарєв В.В.						
Керівн.		Слєсарєв В.В.				НТУ ДП 124; 124м-21		
Н.контр.		Хомяк Т.В.						
Зав. каф.		Желдак Т.А.						

Додаток Б

Відгук
на кваліфікаційну роботу магістра
Студента групи *124м-21-1 Василенко Юрія Анатолійовича*
(група) (ПІБ)
Спеціальності 124 Системний аналіз

Тема кваліфікаційної роботи магістра: *«Аналіз інструментів та технологій створення та керування web–сайтів та CRM–систем».*

Обсяг кваліфікаційної роботи магістра: 54с., 28 рис., 24 табл., 4 додатків, 15 джерел.

Мета кваліфікаційної роботи магістра: запропонувати шляхи та методи покращення економічних показників підприємства в ході впровадження CRM-системи.

Актуальність теми обумовлена високою конкуренцією на інтернет-ринку, адже правильно обрані та оптимізовані веб-проекти мають вищу швидкість відповіді на запит клієнта.

Тема кваліфікаційної роботи магістра безпосередньо пов'язана з об'єктом діяльності магістра спеціальності 124 Системний аналіз, оскільки включає все необхідні етапи системного дослідження, а саме: аналіз об'єкта дослідження, побудову відповідних моделей і алгоритмів і впровадження їх у практичній діяльності.

Виконані в кваліфікаційній роботі магістра завдання **відповідають** вимогам до професійної діяльності фахівця освітньо-кваліфікаційного рівня магістра.

Практичне значення результатів кваліфікаційної роботи магістра полягає в тому, що впровадження сучасних веб-додатків для роботи з клієнтами спричиняє підвищення економічних показників для власника. Важливо правильно аналізувати потреби клієнта та використовувати відповідні інструменти.

Оформлення пояснювальної записки та демонстраційного матеріалу до неї виконано згідно з вимогами. Роботу виконано самостійно, відповідно до завдання та у повному обсязі.

Кваліфікаційна робота магістра в цілому заслуговує на оцінку: добре, а її автор заслуговує на присвоєння кваліфікації «магістр з системного аналізу».

Керівник кваліфікаційної роботи магістра,
Д.т.н., проф. ._____ Слєсарєв В.В

Додаток В

Рецензія
на кваліфікаційну роботу магістра
Студента групи *124м-21- Василенко Юрія Анатолійовича 1*
Спеціальності 124 Системний аналіз

Тема кваліфікаційної роботи: *«Аналіз інструментів та технологій створення та керування web–сайтів та CRM–систем».*

Обсяг кваліфікаційної роботи магістра: 54с., 28 рис., 24 табл., 4 додатків, 15 джерел.

Висновок про відповідність роботи завданню та освітньо-професійній програмі спеціальності – кваліфікаційна робота відповідає вимогам до професійної діяльності фахівця освітньо-кваліфікаційного рівня магістра спеціальності 6.040303 Системний аналіз. .

Зміст пояснювальної записки відповідає темі кваліфікаційної роботи.

Загальна характеристика кваліфікаційної роботи, ступінь використання нормативно-методичної літератури та передового досвіду. Кваліфікаційна робота містить два розділи. В інформаційно-аналітичному розділі розглядаються інструменти створення веб-сайтів та веб-додатків. Зроблений висновок про доцільність користування CRM-системами компаніями малого та середнього бізнесу. У спеціальному розділі описуються популярні CRM-системи, виконується аналіз щодо впровадження CRM-системи згідно потреб клієнта, вибір оптимального варіанта та описується процес розробки CRM-системи для клієнта.

Позитивні сторони **кваліфікаційної роботи** роботи: наведено розширений опис процесу створення веб-сайтів, функцій та можливостей сучасних CRM-систем, порівняння популярних CRM-систем, розроблена реальна CRM-система на мові програмування PHP, яка була передана замовнику. Проведена робота з реальним клієнтом, вимоги якого були задоволені.

Основні недоліки кваліфікаційної роботи: недостатньо математичного аналізу для поставленої задачі, метод аналізу ієрархій виконаний, але результат варто було б порівняти з іншими методами.

Кваліфікаційна робота магістра в цілому заслуговує оцінки: _____, а її автор Дьячков А. О. заслуговує присвоєння кваліфікації “магістр з системного аналізу”.

Рецензент,

Додаток Г

Таблиця Г.1 – Знаходження суми стовбця за критерієм «ціна»

цена	пб	нет	амо	приватная
пб	1,00	7,00	3,00	5,00
нет	0,14	1,00	0,20	0,33
амо	0,33	5,00	1,00	3,00
приватная	0,20	3,00	0,33	1,00
сумм	1,68	16,00	4,53	9,33

Таблиця Г.2 – Знаходження суми стовбця за критерієм «кількість користувачів»

пользователи	пб	нет	амо	приватная
пб	1,00	3,00	0,33	5,00
нет	0,33	1,00	0,20	3,00
амо	3,00	5,00	1,00	7,00
приватная	0,20	0,33	0,14	1,00
сумм	4,53	9,33	1,68	16,00

Таблиця Г.3 – Знаходження суми стовбця за критерієм «вихідний код»

код	пб	нет	амо	приватная
пб	1,00	0,33	0,20	0,14
нет	3,00	1,00	0,33	0,20
амо	5,00	3,00	1,00	0,33
приватная	7,00	5,00	3,00	1,00
сумм	16,00	9,33	4,53	1,68

Таблиця Г.4 – Знаходження суми стовбця за критерієм «чат»

чат	пб	нет	амо	приватная
пб	1,00	0,20	0,33	0,14
нет	5,00	1,00	3,00	0,33
амо	3,00	0,33	1,00	0,20
приватная	7,00	3,00	5,00	1,00
сумм	16,00	4,53	9,33	1,68

Таблиця Г.5 – Знаходження суми стовбця за критерієм «можливість доопрацювання»

доработка	пб	нет	амо	приватная
пб	1,00	0,33	0,20	0,14
нет	3,00	1,00	0,33	0,20
амо	5,00	3,00	1,00	0,33
приватная	7,00	5,00	3,00	1,00
сумм	16,00	9,33	4,53	1,68

Таблиця Г.6 – Знаходження середнього значення критерія «ціна»

цена	пб	нет	амо	приватная	срзнач
пб	0,60	0,44	0,66	0,54	0,56
нет	0,09	0,06	0,04	0,04	0,06
амо	0,20	0,31	0,22	0,32	0,26
приватная	0,12	0,19	0,07	0,11	0,12
сумм	1,00	1,00	1,00	1,00	1,00

Таблиця Г.7 – Знаходження середнього значення критерія «кількість користувачів»

пользователи	пб	нет	амо	приватная	срзнач
пб	0,22	0,32	0,20	0,31	0,26
нет	0,07	0,11	0,12	0,19	0,12
амо	0,66	0,54	0,60	0,44	0,56
приватная	0,04	0,04	0,09	0,06	0,06
0,00	1,00	1,00	1,00	1,00	1,00

Таблиця Г.8 – Знаходження середнього значення критерія «вихідний код»

код	пб	нет	амо	приватная	срзнач
пб	0,06	0,04	0,04	0,09	0,06
нет	0,19	0,11	0,07	0,12	0,12
амо	0,31	0,32	0,22	0,20	0,26
приватная	0,44	0,54	0,66	0,60	0,56
0,00	1,00	1,00	1,00	1,00	1,00

Таблиця Г.9 – Знаходження середнього значення критерія «чат»

чат	пб	нет	амо	приватная	срзнач
пб	0,06	0,04	0,04	0,09	0,06
нет	0,31	0,22	0,32	0,20	0,26
амо	0,19	0,07	0,11	0,12	0,12
приватная	0,44	0,66	0,54	0,60	0,56
сумм	1,00	1,00	1,00	1,00	1,00

Таблиця Г.10 – Знаходження середнього значення критерія «можливість доопрацювання»

доработка	пб	нет	амо	приватная	срзнач
пб	0,06	0,04	0,04	0,09	0,06
нет	0,19	0,11	0,07	0,12	0,12
амо	0,31	0,32	0,22	0,20	0,26
приватная	0,44	0,54	0,66	0,60	0,56
сумм	1,00	1,00	1,00	1,00	1,00

Створюємо матрицю векторів середніх значень критеріїв.

В результаті усіх дій ми маємо матрицю ваг альтернатив по кожному критерію (табл. Г.11) та вектор ваг критеріїв (табл. Г.12):

Таблиця Г.11 – Матриця векторів середніх значень критеріїів

	цена	пользователи	код	чат	доработка
пб	0,56	0,26	0,06	0,06	0,06
нет	0,06	0,12	0,12	0,26	0,12
амо	0,26	0,56	0,26	0,12	0,26
приватная	0,12	0,06	0,56	0,56	0,56

Таблиця Г.12 – Вектор ваг критеріїв

Критерій	Вага
цена	0,16
пользователи	0,10
код	0,06
чат	0,26
доработка	0,42

Помноживши отриману матрицю на стовпець по правилом рядок на стовпець (матрично), отримуємо ваги альтернатив з точки зору досягнення мети (табл. Г.13):

Таблиця Г.13 – Ваги альтернатив

пб	0,16	16%
нет	0,15	15%
амо	0,26	26%
приватная	0,44	44%
сумм	1,00	

Таким чином, альтернатива «Приватна CRM-система» є найбільш привабливою для досягнення нашої мети.