**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ**

**Кафедра економічної кібернетики**

ДОПУСТИТИ ДО ЗАХИСТУ

Завідувач кафедри

економічної кібернетики

\_\_\_\_\_\_\_\_\_\_ Іванченко Н. О.

«\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_2021 р.

**ВИПУСКНА РОБОТА**

(Пояснювальна записка)

Здобувача освітнього ступеня «Бакалавр»

**Тема:** Розробка CRM системи кредитно-фінансової організації

**Виконав:** Гайдучик А.І.

**Керівник:** к.е.н. Густера О.М.

**Консультанти з розділів:**

Розділ 1: к.е.н. Густера О.М.

Розділ 2: к.е.н. Густера О.М.

**Нормоконтролер із ЄСКД (ЄСПД):**

к.е.н., Густера О.М.

Національний авіаційний університет

Факультет економіки та бізнес-адміністрування

Кaфeдpa економічної кібернетики

Освітній ступінь «Бакалавр»

Освітньо-професійна програма «Економічна кібернетика»

 ЗАТВЕРДЖУЮ

 Завідувач кафедри

 економічної кібернетики

 \_\_\_\_\_\_\_\_\_\_\_\_Іванченко Н.О.

 «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2021 р.

**ЗАВДАННЯ НА ВИПУСКНУ РОБОТУ**

Студента: Гайдучика Андрія Івановича

Тема роботи: Розробка CRM системи кредитно-фінансової організації затверджена наказом ректора No 431/ст від 18.03.2021 р.

1. Термін здачі студентом закінченої роботи на кафедру:

2. Вихідні дані до роботи: наукові роботи в галузі розробки CRM систем зарубіжних та вітчизняних вчених.

3. Зміст дослідження:

* дослідження особливостей архітектури та практичного використання веб-фреймворків;
* дослідження особливостей архітектури та практичного використання CRM-систем;
* постановка задачі розробки веб-додатку CRM-системи кредитно-фінансової організації;
* розробка веб-додатку CRM-системи кредитно-фінансової організації, аналіз результатів розробки та розробка рекомендацій щодо практичного використання веб-додатку CRM-системи кредитно-фінансової організації.

4. Перелік обов‘язкових демонстраційних матеріалів: 12 слайдів

**Календарний план**

|  |  |  |  |
| --- | --- | --- | --- |
| No п/п | Назва етапів випускної роботи | Термін виконання | Позначки керівника про виконання завдань |
| 1 | Отримання завдання на випускну роботу | 17.05.2021 |  |
| 2 | Огляд літератури за темою | 18.05.2021 |  |
| 3 | Сутність та галузі веб-фреймворків | 19.05.2021-20.05.2021 |  |
| 4 | Сутність, галузі використання та особливості архітектури CRM-систем | 21.05.2021 |  |
| 5 | Постановка задачі розробки веб-додатку | 22.05.2021 |  |
| 6 | Розробка веб-додатку | 23.05.2021-25.05.2021 |  |
| 7 | Aнaлiз oтpимaних peзультaтiв | 26.05.2021 |  |
| 8 | Розробка слайдів та написання доповіді | 27.05.2021 |  |
| 9 | Попередній захист випускної роботи | 28.05.2021 |  |
| 10 | Корегування роботи за результатами попереднього захисту | 29.06.2021 |  |
| 11 | Остаточне оформлення випускної роботи та слайдів | 01.06.2021 |  |
| 12 | Підписання відгуку та рецензії | 04.06.2021-08.06.2021 |  |
| 13 | Захист випускної роботи у ДЕК | 17-18.06.2021 |  |

5. Дата видачі завдання:

Керівник:

к.е.н. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Густера О.М.

Завдання прийняв для виконання \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Гайдучик А.І.

**РЕФЕРАТ**

**Гайдучик Андрій Іванович. Розробка CRM системи кредитно-фінансової організації.** – Випускна робота бакалавра зі спеціальності 051 «Економіка», ОПП «Економічна кібернетика». Національний авіаційний університет Міністерства освіти і науки України, м. Київ, 2021.

Випускна робота містить 65 сторінок, 21 рисунок, список використаних джерел з 40 найменувань та 2 додатки.

Об’єктом дослідження даної випускної роботи виступають сучасні CRM-системи для кредитно-фінансових організацій.

Предметом дослідження даної випускної роботи виступає процес розробки CRM-системи для кредитно-фінансової організації засобами фреймворка Django.

Метою даної випускної роботи є розробка CRM-системи для кредитно-фінансової організації засобами фреймворка Django, яка полегшить не тільки роботу CRM-менеджерів компанії, завдяки своєму простиму інтерфейсу, але і клієнтів, які зможуть легко відслідковувати інформацію в особистому кабінеті.

Наукова новизна дослідження полягає у розробці CRM-системи для кредитно-фінансової організації засобами фреймворка Django, яка підвищує ефективність роботи CRM-менеджерів компанії, завдяки своєму простиму інтерфейсу, та клієнтів, які зможуть легко відслідковувати інформацію в особистому кабінеті.

При написанні роботи використовувалися методи дослідження: системний підхід, методи аналізу і синтезу, аналогії та порівняльного аналізу.

***Ключові слова:*** *веб-додаток, фреймворк, веб-фреймворк, CRM система, фінансово-кредитна організація.*

**ABSTRACT**

**Gaiduchyk Andriy Ivanovych. Development of CRM system of credit and financial organization.**  - Bachelor graduation paper in the specialty 051 "Economics", "Economic Cybernetics". National Aviation University of the Ministry of Education and Science of Ukraine, Kyiv, 2021.

The graduation paper contains 65 pages, 21 figures, a list of references consisting of 35 items, and 2 Appendix.

The object of study of this dissertation are modern CRM-systems for credit and financial organizations.

The subject of research of this final work is the process of developing a CRM-system for credit and financial organization by means of the Django framework.

The purpose of this thesis is to develop a CRM-system for credit and financial organization through the Django framework, which will facilitate not only the work of CRM-managers of the company, thanks to its simple interface, but also customers who can easily track information in the personal account.

The scientific novelty of the study is to develop a CRM-system for credit and financial organization using the Django framework, which increases the efficiency of CRM-managers of the company, thanks to its simple interface, and customers who can easily track information in the personal account.

In writing the research methods were used: a systematic approach, methods of analysis and synthesis, analogy and comparative analysis.

***Keywords****: web application, framework, web framework, CRM system, financial and credit organization.*

**ЗМІСТ**

[ВСТУП 7](#_Toc74648651)

[РОЗДІЛ 1. ВЕБ-ФРЕЙМВОРКИ 9](#_Toc74648652)

[1.1. Загальна інформація про веб-фреймворки 9](#_Toc74648653)

[1.2. Особливості веб-фреймворків 12](#_Toc74648654)

[1.3. Архітектура веб-додатків 14](#_Toc74648655)

[Висновки за розділом 1 19](#_Toc74648656)

[РОЗДІЛ 2. CRM-СИСТЕМИ. 21](#_Toc74648657)

[2.1. Загальна інформація про CRM-системи. 21](#_Toc74648658)

[2.2. Архітектура CRM-систем 26](#_Toc74648659)

[2.3. Особливості розробки та впровадження CRM систем 28](#_Toc74648660)

[Висновки за розділом 2 33](#_Toc74648661)

[РОЗДІЛ 3. РОЗРОБКА ВЕБ-ПРОЕКТУ 35](#_Toc74648662)

[3.1. Постановка завдання 35](#_Toc74648663)

[3.2. Вибір інструментів для реалізації 35](#_Toc74648664)

[3.3 Програмна реалізація веб-проекту 40](#_Toc74648665)

[3.4 Опис функціональності 46](#_Toc74648666)

[Висновки за розділом ІІІ 55](#_Toc74648667)

[ВИСНОВКИ 57](#_Toc74648668)

[СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ 59](#_Toc74648669)

[ДОДАТКИ 62](#_Toc74648670)

# ВСТУП

Сучасні умови розвитку прикладних інформаційних систем вимагають постійного вдосконалення та адаптації до вимог клієнтів. Саме клієнтоорієнтовані CRM-системи є найбільш перспективними та актуальними для сучасного бізнесу.

На сьогодні, конкуренція на всіх рівнях ринку досить висока. Компаніям, щоб мати перевагу в конкурентній боротьбі необхідно не лише залучати нових клієнтів, а й утримувати вже існуючих. Проблема утримання та розвитку існуючих клієнтів є інколи більш актуальною, ніж залучення нових.

Щоб втримати клієнта необхідно враховувати його побажання та інтереси. Такий підхід до ведення бізнесу називається клієнтоорієнтованим. Проте за великого об’єму клієнтської бази врахування інтересів кожного клієнта є складно доступною метою.

Рішенням даної проблеми є впровадження CRM системи. Ця абревіатура розшифровується як Customer Relationship Management, що в перекладі українською означає «управління взаємовідносин з клієнтами».

CRM-модель взаємодії визначає, що центром всієї філософії бізнесу є клієнт, а основними напрямками діяльності є заходи з підтримки ефективного маркетингу, продажів і обслуговування клієнтів. Підтримка цих бізнес-цілей включає збір, зберігання і аналіз інформації про споживачів, постачальників, партнерів, а також про внутрішні процеси компанії. Функції для підтримки цих бізнес-цілей включають продажі, маркетинг, підтримку споживачів і взаємовідносини клієнт - управління.

Метою даної випускної роботи є розробка CRM-системи для кредитно-фінансової організації засобами фреймворка Django, яка полегшить не тільки роботу CRM-менеджерів компанії, завдяки своєму простиму інтерфейсу, але і клієнтів, які зможуть легко відслідковувати інформацію в особистому кабінеті.

Об’єктом дослідження даної випускної роботи виступають сучасні CRM-системи для кредитно-фінансових організацій.

Предметом дослідження даної випускної роботи виступає процес розробки CRM-системи для кредитно-фінансової організації засобами фреймворка Django.

Випускна кваліфікаційна робота складається з вступу, трьох розділів і висновку.

У першому розділі описані фреймворки, їх типи, архітектура і особливості практичного використання в сучасних умовах цифрової економіки.

У другому розділі розглянуті CRM-системи в цілому. Також детально проаналізовано особливості розвитку та застосування сучасних систем у бізнесі, їх особливості та архітектура.

 У третьому розділі розглянуто етапи практичної розробки веб-проекту - постановка завдання та засоби для розробки, вказані плюси даних засобів, описана функціональність даного веб-додатку. Також наведено практичні рекомендації щодо застосування отриманих результатів на сучасних підприємствах в умовах цифрової економіки.

# РОЗДІЛ 1. ВЕБ-ФРЕЙМВОРКИ

# 1.1. Загальна інформація про веб-фреймворки

Фреймворк веб-програми або веб-фреймворк - це програмне забезпечення, яке створене для підтримки розробки динамічних веб-сайтів, веб-сервісів та веб-додатків. Веб-фреймворк (WF) підтримує створення, розробку та публікацію веб-додатків та веб-сайтів. Це може включати веб-сервіси, API та інші ресурси. Веб-фреймворки - це фреймворки, які пропонують стандартний та доступний спосіб створення та розробки веб-додатків.

Незважаючи на те, що веб-програми є звичними явищами в Інтернеті і використовуються щодня більшістю підключених до Інтернету людей, вони все ще перебувають у зародковому стані, коли йдеться про розробку інструментів. Ось чому так важливо для сучасних підприємства важливо мати веб-фреймворк, який би виступав системою підтримки для розробки та розгортання таких додатків. Фреймворк веб-додатків є одним із таких інструментів, і у даній роботі буде показано, як це може допомогти створити веб-програму, яка відповідає найсучаснішим веб-стандартам.

Веб-фреймворк - це бібліотека коду, яка робить веб-розробку швидшою та простішою, надаючи основні шаблони для створення надійних, масштабованих та ремонтопридатних веб-додатків. Веб-фреймворки існують, щоб спростити розробнику створення веб-програми. Це можна представити як спосіб створення ярликів, які можуть запобігти переважному і повторюваному коду. Наприклад, якщо написано код для обробки перевірки даних для певної веб-програми. Щоб запобігти необхідності переписувати цей код з нуля кожного разу, коли створюється веб-сайт або веб-служба, веб-фреймворк робить це самостійно.

Веб-фреймворк відрізняється від веб-сервера тим, що веб-сервер насправді запускає веб-додаток, тоді як веб-фреймворк більше нагадує віртуальну базу даних або бібліотеку, яка допомагає пришвидшити процес розробки та написання.

Фреймворки надають функціональність у своєму коді або через розширення для завершення повсякденних операцій, необхідних для запуску веб-додатків. Ці операції включають:

* маршрутизація URL-адрес,
* управління та перевірка форми введення,
* налаштування HTML, XML, JSON та інших продуктів із механізмом шаблонування,
* конфігурація з'єднання з базою даних та рішуча маніпуляція даними за допомогою об'єктно-реляційного картографування (ORM) ,
* веб-безпека від міжсайтової підробки (CSRF), SQL Injection, міжсайтових сценаріїв (XSS) та інших частих шкідливих атак,
* сховище сесії та пошук.

Існує кілька веб-фреймворків, розроблених для того, щоб суттєво спростити кодування та розробку додатків для тих, хто працює в цій галузі. Ці фреймворки допомагають кодам, таким як HTML, Ruby та Python, коли мова заходить про створення веб-додатків та веб-служб. Існує велика кількість задач, які можна виконати за допомогою веб-фреймворку, наприклад, генерація HTML, формування шаблонів за замовчуванням та можливість доступу до бібліотеки інформації про кодування, щоб зробити код простим та автоматизованим кожного разу, коли розробник робить новий веб-додаток.

Рекомендується використовувати фреймворк, який допоможе впорядкувати та пришвидшити процес, а успішні фреймворки, такі як Django, розроблені, щоб стати потужним фактором розвитку для проекту. Є певні особливості, які слід враховувати при використанні веб-фреймворків для написання коду.

* Взаємодія з базами даних - Багато веб-фреймворків мають можливість застосовувати взаємодію з базами даних у всеосяжному пакеті.
* Генерація HTML - фреймворки мають можливість генерувати HTML-коди за допомогою шаблонів, що може полегшити розробникам використання під час встановлення значень та встановлення стандартів.
* Маршрутизація - Більшість фреймворків можуть обробляти можливість зіставити URL-адресу, яку запитує користувач, з кодом, який відповідає їй.
* Веб-фреймворки допомагають виконувати різноманітні завдання, починаючи від створення шаблонів та доступу до бази даних, до управління сеансами та повторного використання коду. Це великий актив для створення динамічних веб-сайтів, але може бути використаний і для більш статичних веб-сайтів.
* Веб-фреймворк - інструмент, який полегшує процес написання і запуску веб-додатку. Використовуючи фреймворк не потрібно самостійно писати велику кількість коду і витрачати час на пошук потенційних прорахунків і помилок [3].

На початку ери веб-розробки всі додатки писалися вручну, і тільки розробник додатку міг змінити або розгорнути. Веб-фреймворки дозволили уникнути цього. З 1995 року всі проблеми, пов'язані зі зміною структури додатка, були в тій чи іншій мірі усунені завдяки появі загального підходу до розробки веб-додатків. У цей час з'явилися мови для вебу. Зараз їх різноманітність дозволяє вибрати відповідний як для статичних, так і для динамічних сторінок. Залежно від поставленого завдання, можна вибрати один фреймворк, що покриває всі потреби, або поєднати кілька.

Далі розглянемо основні типи веб-фреймворків. У веб-фреймворків є дві основні функції: робота на серверній стороні (бекенд) і робота на клієнтській стороні (фронтенд).

Фронтенд-фреймворки пов'язані з зовнішньою частиною програми. Простими словами, вони відповідають за зовнішній вигляд програми. Бекенд відповідає за внутрішні налаштування додатку.

Серверні веб-фреймворки. Правила та архітектура таких веб-фреймворків не дає можливості створити веб-додаток з багатим інтерфейсом. Вони обмежені в своїй функціональності, проте з їх допомогою всеодно можна створювати прості сторінки і різні форми. Також вони можуть формувати вихідні дані і відповідати за безпеку в разі атак. Все це суттєво може спростити процес розробки. Серверні веб-фреймворки в основному відповідають за окремі, але критично важливі частини програми, без яких вона не зможе нормально працювати. Кілька найпопулярніших серверних фреймворків і мови, з якими вони працюють:

* Django - Python;
* Zend - PHP;
* Express.js - JavaScript;
* Ruby on Rails -Ruby.

Клієнтські веб-фреймворки. На відміну від серверних, клієнтські веб-фреймворки ніяк не пов'язані з логікою програми. Цей тип веб-фреймворків працює в браузері. З їх допомогою можна поліпшити і впровадити нові, призначені для користувача інтерфейси.

Фронтенд-фреймворки дозволяють створювати різні анімації і односторінкові додатки. Всі клієнтські фреймворки відрізняються по функціональності і використанню. Ось деякі з них:

* Backbone + Marionette;
* Angular;
* Ember.js;
* Vue.js.

Всі ці веб-фреймворки використовують JavaScript.

# 1.2. Особливості веб-фреймворків

Архітектура будь-якого веб-фреймворку містить три складові, а саме «Модель-Подання-Контролер» (Model-View-Controller) (MVC).

MVC - схема поділу даних програми, користувацького інтерфейсу і керуючої логіки на три окремих компоненти таким чином, що модифікація кожного компонента може здійснюватися незалежно. Структура MVC показана на рисунку 1.



Рис. 1.1. Структура MVC

Модель (Model) надає дані і реагує на команди контролера, змінюючи свій стан. Подання (View) відповідає за відображення даних моделі користувачу, реагуючи на зміни моделі. Контролер (Controller) інтерпретує дії користувача, сповіщаючи модель про необхідність змін.

Особливості веб-фреймворків:

1. Веб-кешування. Кешування просто допомагає зберігати різні документи і дозволяє уникнути набридливого перезавантаження сервера. Користувачі можуть використовувати його в різних системах при певних умовах. Також воно працює на серверній стороні. Наприклад, можна помітити кешування веб-сторінки на сторінці результатів пошукової видачі Google.

2. Скаффолдинг. Це ще одна технологія, підтримувана певними MVC-фреймворками. Фреймворк може автоматично згенерувати типові частини програми або навіть всю структуру проекту (якщо мова іде про ініціалізацію). Це дозволяє істотно збільшити швидкість розробки і стандартизує кодову базу.

3. Система веб-шаблонів. Система веб-шаблонів являє собою набір різних методологій і програмного забезпечення, реалізованих для створення і розгортання веб-сторінок. Для обробки веб-шаблонів використовуються шаблонізатор. Вони є інструментом фреймворка, що відповідає за веб-публікацію.

4. Зіставлення URL. Якщо необхідно спростити індексацію сайту пошуковими системами, в той же час створюючи привабливу назву для сайту, то використовується ця функція фреймворків. Також зіставлення URL може полегшити доступ до сторінок сайту.

5. Додатки. Безліч типів веб-додатків підтримуються веб-фреймворками. В основному вони застосовуються для створення таких додатків, як блоги, форуми, CMS і т. д. Вся ця функціональність властива всім веб-фреймворкам. Однак, як не парадоксально, з таким широким асортиментом розробник розгублюється і не може нічого вибрати. Тому потрібно придумати критерії, згідно яких можна вибрати найкращий інструмент, який полегшує розробку. Наприклад, вибір може залежати від того, якій мові програмування надається перевага. Крім того, потрібно звернути увагу на асортимент можливостей набору інструментів фреймворка. Говорячи про переваги, варто відзначити, що вони можуть служити як на благо, так і на шкоду. Звичайно, краще використовувати фреймворки, які простіше вивчити.

# 1.3. Архітектура веб-додатків

Архітектура веб-додатків - це «скелет» або макет, що відображає взаємодію між компонентами програми, системами проміжного програмного забезпечення, користувальницькими інтерфейсами та базами даних. Цей тип взаємодії дозволяє одночасно працювати декільком програмам. Як тільки користувач відкриває веб-сторінку, сервер надсилає певні дані браузеру як відповідь на запит користувача. Веб-клієнт (або агент користувача) може запитувати веб-ресурси або більш відомі веб-документи (HTML, JSON, PDF тощо) через веб-сервер. Після цього починається взаємодія між користувачем та веб-сайтом. Архітектура програмного забезпечення - це скелет та всі компоненти високого рівня системи та те, як вони взаємодіють між собою. Він відповідає на запитання «Де?» і «Як?».

Завдяки архітектурі програмного забезпечення стає легше побачити загальну картину. Головна мета - визначити як функціональні вимоги, так і вимоги до якості та обробити їх, щоб поліпшити загальну якість програми. Отже, загалом, за допомогою архітектури програмного забезпечення можна контролювати продуктивність, масштабованість та надійність. Щодо дизайну програмного забезпечення, то тут більше йдеться про дизайн рівня коду, і він відповідає за функціональність кожного модуля та його цілі.

За допомогою дизайну програмісти мають ефективну загальну мову для пошуку рішень для повторюваних питань та їх концептуалізації. Це мінімізує обсяг роботи, яку вони виконують, оскільки немає необхідності винаходити колесо заново.

DNS або система доменних імен - це фундаментальна система, яка допомагає шукати доменне ім'я та IP-адресу, і таким чином певний сервер отримує запит, надісланий користувачем. Можна сказати, що DNS схожий на телефонну книгу, але для веб-сайтів в Інтернеті.

Load Balancer в основному займається горизонтальним масштабуванням. За допомогою перенаправлення вхідних запитів на один із декількох серверів, балансування навантаження надсилає відповідь користувачеві. Зазвичай сервери веб-додатків існують у вигляді кількох копій, що відображають один одного. Отже, будь-який сервер обробляє запити однаковим чином, а балансир навантаження розподіляє між ними завдання, щоб вони не переповнювались/перенавантажувались.

Сервери веб-додатків обробляють запити користувачів і відправляють документи (JSON, XMK ​​тощо) назад у браузер. Для виконання цього завдання, як правило, йдеться про внутрішні інфраструктури, такі як база даних, сервер кешування, черга завдань та інші. Крім того, щонайменше два сервери, підключені до балансувача навантаження, управляють обробкою запитів користувача.

База даних надає інструменти для організації, додавання, пошуку, оновлення, видалення та виконання обчислень. У більшості випадків сервери веб-додатків безпосередньо взаємодіють із серверами завдань.

Сервіс кешування забезпечує зберігання даних, що дозволяє зберігати та шукати дані. Щоразу, коли користувач отримує якусь інформацію з сервера, результати цієї операції надходять у кеш. Отже, майбутні запити повертаються швидше. Таким чином, кешування дозволяє посилатися на попередній результат, щоб зробити обчислення набагато швидше. Тому кешування ефективне, коли:

* обчислення повільні;
* обчислення, ймовірно, відбуватимуться багато разів;
* коли результати однакові для конкретного запиту.

Черга завдань складається з двох компонентів: самої черги завдань та серверів. Ці сервери обробляють завдання в черзі. Трапляється, що більшість веб-серверів повинні виконувати величезну кількість завдань, які не мають першочергового значення. Тому, коли завдання потрібно виконати, воно потрапляє до черги завдань і експлуатується за графіком.

Багато веб-додатків підтримують пошук за допомогою текстової функції або так званого запиту, а потім програма надсилає користувачеві найбільш релевантні результати. Ця технологія отримала назву служби повнотекстового пошуку. За допомогою ключових слів він здійснює пошук необхідних даних серед величезної кількості документів.

Коли веб-програма досягає певного рівня, служби створюються у вигляді окремих програм. Вони не так помітні серед інших компонентів веб-додатків, але веб-програма та інші служби взаємодіють з ними.

Майже кожен сучасний додаток передбачає роботу з такими даними, як збір, зберігання та аналіз. Ці процеси вимагають трьох стадій:

* Дані надсилаються до даних “firehose”, який забезпечує потоковий інтерфейс для поглинання та обробки даних.
* Сирі, оброблені та додаткові дані надсилаються у хмарне сховище.
* Оброблені та додаткові дані також надходять у сховище даних.

CDN або система доставки вмісту займається надсиланням HTML-файлів, CSS-файлів, файлів JavaScript та зображень. Він забезпечує вміст кінцевого сервера по всьому світу, тому люди можуть завантажувати різні джерела.

Існує лише три моделі компонентів веб-додатків. Це тісно пов’язано з кількістю служб та баз даних, що використовуються для веб-додатків.

Один веб-сервер, одна база даних. Особливістю цього сервера є те, що він використовує як один сервер, так і єдину базу даних. Це робить модель найменш надійною. Як тільки сервер виходить з ладу, така модель також припиняє працювати. Отже, ця модель не використовується зазвичай для побудови веб-додатків. Тим не менше, вона досить часто використовується для запуску тестових проектів та вивчення та розуміння основ веб-додатків.

Кілька веб-серверів, одна база даних. Ця модель сучасного дизайну веб-додатків має досить цікаву особливість: вона не зберігає жодних даних. Коли клієнт передає інформацію веб-серверу, він обробляє та записує її в базу даних, але управління цими даними відбувається поза сервером. Це називається архітектурою без громадянства. Для роботи з цією моделлю розробникам потрібні принаймні два веб-сервери. Це важливо для того, щоб зробити модель надійнішою, оскільки якщо один сервер не працює, інший прийме на себе відповідальність за роботу системи. Отже, у такому випадку всі запити автоматично надходять на новий сервер, не впливаючи на функціонування веб-програми. Таким чином, ця модель надійніша, ніж окремий сервер. Однак, якщо щось трапиться з базою даних, програма вийде з ладу.

Кілька веб-серверів, кілька баз даних. Це найефективніша та найнадійніша модель веб-додатків. Причина полягає в тому, що і сервери, і бази даних мають кілька замін. Отже, у разі відмови є два виходи: зберігати дані у всіх доступних базах даних або розподіляти їх рівномірно між ними. У будь-якому випадку, веб-сайт буде цілим і надійним.

Служби програм (мікросервіси та безсерверні), як правило, гнучкі, оскільки спрощують оновлення та масштабування. Застосування цієї моделі дозволяє розділити веб-сервери на менші частини: «послуги» в мікросервісах та «функції» в безсерверних. Таким чином, модифікувати та масштабувати самостійно, використовуючи кожен з них, простіше.

Односторінкові програми або SPA-програми мають на меті полегшити роботу програмістів. Щоб здійснити дію на веб-сайті, користувачеві не потрібно щоразу завантажувати абсолютно нові сторінки. Натомість вони можуть взаємодіяти з ним, отримуючи оновлений вміст на поточній сторінці.

Цей тип архітектурного веб-дизайну створений таким чином, що він вимагає найнеобхідніший вміст та дані. Таким чином, SPA-центри запобігають перервам в діяльності користувачів, щоб покращити інтуїтивний та інтерактивний досвід користувачів. JavaScript є найпоширенішою мовою програмування в цьому типі архітектури.

Server Side Rendering (SSR) - це процес надання клієнтському веб-сайту Javascript фреймворку HTML та CSS на сервері. За допомогою цього інструменту можна швидко доставити найважливіші ресурси, тим самим прискорюючи швидкість браузера, надання сторінки користувачеві займає менше часу.

При створенні додатка SSR, сервер компілює всі дані та подає новий документ HTML для кожного запиту. І як тільки браузер отримає CSS, він зможе розфарбувати інтерфейс, і не потрібно чекати завантаження JavaScript. Таким чином це пришвидшує завантаження сторінки.

Мікросервіси - це один із типів SOA (архітектура, орієнтована на послугах). Загалом, мікросервіси займаються малими та легкими послугами та виконують єдину функціональність. Цей тип веб-архітектури є ефективним і продуктивним. За допомогою цього розробники можуть заощадити багато часу.

Оскільки компоненти мікросервісів не обов’язково будуються з однією мовою програмування, вони не взаємозалежні. Це означає, що розробники можуть вільно обирати технологію, яка їм більше подобається, що в результаті робить розвиток архітектури мікросервісів швидшим та простішим.

Безсерверна архітектура означає, що налаштування та адміністрування серверів, на яких запущено програмне забезпечення для управління, більше не є необхідністю. Але вся інфраструктура підтримується сторонніми провайдерами. Сторонній постачальник вносить внесок у аутсорсинг серверів та управління інфраструктурою.

У цьому типі архітектури веб-додатків розробник додатків звертається до стороннього постачальника послуг хмарної інфраструктури щодо сервера аутсорсингу, а також управління інфраструктурою.

Основні критерії побудови надійної архітектури додатків:

* ефективність,
* гнучкість,
* багаторазовість використання,
* легко перевіряється,
* вирішує проблеми послідовно і успішно,
* з добре структурованим і зрозумілим кодом,
* масштабована в процесі розробки,
* має високу швидкість відгуку,
* не має жодної точки відмови,
* проста,
* використовує стандартні стандарти безпеки.

Ефективність використання веб-додатків тісно пов'язана з їх архітектурою.

# Висновки за розділом 1

На відміну від серверних, клієнтські веб-фреймворки ніяк не пов'язані з логікою програми. Цей тип веб-фреймворків працює в браузері. З їх допомогою можна поліпшити і впровадити нові, призначені для користувача інтерфейси.

Фронтенд-фреймворки дозволяють створювати різні анімації і односторінкові додатки. Всі клієнтські фреймворки відрізняються по функціональності і використанню.

Рекомендується використовувати фреймворк, який допоможе впорядкувати та пришвидшити процес, а успішні фреймворки, такі як Django, розроблені, щоб стати потужним фактором розвитку для проекту. Є певні особливості, які слід враховувати при використанні веб-фреймворків для написання коду.

Незважаючи на те, що веб-програми є звичними явищами в Інтернеті і використовуються щодня більшістю підключених до Інтернету людей, вони все ще перебувають у зародковому стані, коли йдеться про розробку інструментів. Ось чому так важливо для сучасних підприємства важливо мати веб-фреймворк, який би виступав системою підтримки для розробки та розгортання таких додатків. Фреймворк веб-додатків є одним із таких інструментів, і у даній роботі буде показано, як це може допомогти створити веб-програму, яка відповідає найсучаснішим веб-стандартам.

# РОЗДІЛ 2. CRM-СИСТЕМИ.

# 2.1. Загальна інформація про CRM-системи.

CRM розшифровується як «Управління відносинами з клієнтами». CRM - це програмне забезпечення для бізнесу, призначене для управління взаємодією компанії зі своїми клієнтами та потенційними клієнтами.

Зазвичай CRM-система, яка найчастіше використовується спеціалістами з продажу та маркетингу, допомагає організаціям впорядкувати свій робочий процес, використовуючи автоматизацію продажів (SFA), покращуючи ефективність та продуктивність, стимулюючи ріст бізнесу та збільшуючи задоволеність клієнтів.

Система CRM, що використовується в основному для управління контактами та управління продажами, забезпечує єдиний центр, який зберігає всі дані для ділових контактів. Це може включати клієнтів, потенційних клієнтів, постачальників, партнерів, пресу та колег.

З часом система буде створювати історію взаємодій з кожним контактом. Це може включати електронні листи, дзвінки, нотатки про зустрічі та розпродажі. Перевага такого підходу полягає в тому, що все знаходиться в одному місці, забезпечуючи чітку видимість каналу продажів.

Програма CRM також може бути інтегрована з іншими програмами та каналами, які використовуються, такими як веб-сайт, інструмент цифрового маркетингу та бухгалтерське програмне забезпечення. Це дозволяє створювати автоматизовані робочі процеси, замінюючи рутинні ручні процеси у бізнесі.

Наприклад, можна використовувати CRM-програмне забезпечення для отримання нового запиту з веб-сайту, який ініціює автоматизований процес продажу. Запит спричинить новий запис у CRM, створить завдання для команди продажів і відправить контакту привітальне повідомлення.

Можна визначити наступні функціональні особливості програмного забезпечення CRM:

* орієнтується на клієнта,
* допомагає миттєво знайти кожного клієнта, контакт,
* покращує післяпродажне обслуговування та підтримку,
* підвищує лояльність та утримання клієнтів,
* збільшує продажі та прибутковість,
* класифікує та визначає пріоритети потенційних клієнтів,
* відстежує потенційних клієнтів,
* автоматизує процес продажу,
* керує завданнями та призначеннями,
* створює централізований центр з усім в одному місці,
* створює загальну видимість, яка покладає край дублюванню,
* посилює співпрацю між членами команди та між командами,
* сприяє віддаленій роботі,
* впорядковує процеси, інтегруючи інші системи, які використовуються,
* забезпечує аналіз та звітність,
* точні, докладні звіти та прогнози,
* аналіз діяльності клієнтів,
* управління маркетинговою рентабельністю інвестицій.

Кожен член організації може бачити всі взаємодії з контактом у системі, включаючи їхні зв’язки, що вони придбали та коли, що заплатили, а також багато іншої інформації.

Доклавши всіх необхідних зусиль, щоб зібрати інформацію про потенційного клієнта чи клієнта, продавці можуть в кінцевому підсумку отримати свою інформацію в кількох джерелах - від наліпок до журналів дзвінків. Використання CRM допомагає об’єднати все це у централізованому концентраторі. Програмне забезпечення сприяє співпраці команд, віддаленій роботі та підвищує ефективність. Результат - відсутність дублювання зусиль, відсутність втрачених повідомлень або забутих зворотних дзвінків. Це призводить до більш своєчасного спілкування, поліпшення відносин і, по суті, збільшення продажів.

Система також допоможе керувати командами, контролювати безпеку даних та підтримувати дотримання правил конфіденційності даних.

Можна виділити наступні основні переваги CRM для бізнесу

* покращує управління контактами,
* покращує утримання та задоволення клієнтів,
* покращує управління продажами,
* підвищує ефективність та продуктивність,
* створює організаційну та командну співпрацю,
* сприяє точності звітності та прогнозування продажів,
* покращує розуміння даних щодо маркетингу та розробки продуктів,
* підтримує поглиблені показники продажів,
* підсилює зростання бізнесу.

Щоб максимізувати вигоду, потрібно вбудувати свою CRM-стратегію у бізнес-стратегію. Це створить структуру та організацію для компанії, що сприятиме підвищенню ефективності, продуктивності та прибутковості.

Хоча на початку система була розроблена для команд продажів, переваги використання CRM у всій організації можуть бути досить суттєвими. Це стає все більш важливим для маркетологів та для команд обслуговування клієнтів, що керують післяпродажними комунікаціями.

Програмне забезпечення дає більше розуміння того, як працює бізнес, використовуючи фактичні дані про продажі. Це надає повний огляд клієнтів та їх діяльності, дозволяючи створити справді орієнтований на клієнта фокус, який сформує стратегію взаємодії з клієнтами.

Програмне забезпечення для управління взаємовідносинами з клієнтами є актуальним для багатьох бізнес-функцій та галузевих секторів, включаючи некомерційний сектор та членські організації.

CRM-програмне забезпечення найчастіше використовується для функцій служби продажів, маркетингу та обслуговування. Однак використання в інших командах та керівництві також може бути доречним.

Продавці можуть використовувати CRM для моніторингу нових потенційних клієнтів у своєму каналі продажів, реєстрації дзвінків та приміток до зустрічей, встановлення завдань для подальшого спостереження з клієнтами, створення та надсилання пропозицій та рахунків-фактур. Запускаючи звіти та прогнози, менеджери з продажу мають змогу бачити ефективність своєї команди в реальному часі щодо своїх цілей.

Маркетологи можуть інтегрувати форми своїх веб-сайтів із системою для автоматичного залучення нових потенційних клієнтів та передачі їх команді продажів для подальшого спостереження. Вони можуть надсилати автоматичні електронні листи, щоб привітати нових клієнтів та відстежувати їх відповіді.

Запуск маркетингових кампаній із CRM означає, що вся діяльність клієнтів реєструється у системі. Це дає змогу зрозуміти їх поведінку та провести аналіз рентабельності інвестицій.

CRM із інтегрованою системою довідкової служби означає, що також можна контролювати взаємодію клієнтів після продажу. Будь-які підняті питання підтримки будуть враховані, тому команда продажів може бути попереджена про будь-які потенційні проблеми.

Організації, що ведуть бізнес до бізнесу (B2B), як правило, є цільовою аудиторією для постачальників CRM, але програмне забезпечення також може мати значення для інших ринків.

Як правило, ціна системи визначається кількістю користувачів, кількістю записів клієнтів, вимогами до зберігання та складністю функцій. Більшість постачальників пропонують безкоштовну пробну версію CRM, щоб потенційні користувачі могли побачити, як система працюватиме.

У деяких галузях промисловості можна керувати тисячами клієнтів з коротким циклом продажів або кількома повторними продажами. З іншими може бути лише кілька сотень контактів, але з тривалим циклом продажів, який потребує ретельного відстеження. В обох випадках система допомагає відстежувати взаємодію та контролювати поведінку клієнтів. Це означає, що можна ефективно керувати частотою взаємодій, щоб максимізувати продажі.

Базова система CRM включає управління контактами та управління можливостями. Це дозволяє миттєво знаходити клієнтів та попередні зв’язки та відстежувати потенційних клієнтів. Однак існує безліч маленьких додатків, які можуть суттєво змінити спосіб використання CRM. Наприклад можна налаштувати за власними полями та налаштуваннями, щоб відображати конкретний процес продажу. Бажано включати інтеграцію електронної пошти. Це автоматично захоплюватиме електронні листи до контактів та від них, надаючи повну історію спілкування.

Більшість CRM-систем малого бізнесу не включають багато способів складання звітів, якщо такі є. Найкращі системи включатимуть розробника звітів з прогнозу продажів, а також розробника звітів для спеціальної діяльності.

На сучасному ринку, навіть для малого бізнесу, автоматизація CRM стає необхідною. Автоматизація допомагає підвищити ефективність та продуктивність за рахунок автоматизації ручних процесів. Різні дії в CRM запускають робочі процеси для виконання завдання відповідно до набору правил, який визначаються. Це може означати активацію процесу виставлення рахунків при закритті продажу або нагадування електронною поштою про закінчення терміну гарантії.

Також можна використовувати автоматизацію та інтеграцію для передачі даних між різними системами, які використовуються, оновлення та синхронізації інформації про клієнтів. Тут можна використовувати власну інтеграцію через API, або використовувати платформу інтеграції, таку як Zapier.

CRM з інтегрованим маркетингом дозволяє запускати електронні кампанії безпосередньо з системи та реєструвати залучення контактів до їх облікового запису. З часом накопичується цінна інформація про їх поведінку та інтереси, яка дозволить краще задовольнити їх потреби в майбутньому.

Сучасні Cloud CRM мають певні переваги у порівнянні з локальними версчями. Оскільки хмарне CRM-програмне забезпечення розміщується та підтримується постачальником, воно не вимагає капітальних витрат та обслуговування клієнта. Запустивши тисячі клієнтських CRM-систем, постачальник може створити велику економію від масштабу. Ці фактори роблять хмарну систему дуже доступною з можливістю оплати щомісяця або щороку, тому фінансовий ризик низький.

Інша велика перевага полягає в тому, що користувачі CRM можуть отримати доступ до системи з будь-якого місця через свій Інтернет-браузер. Це може бути вдома, в їх офісі або в дорозі з смартфону.

# 2.2. Архітектура CRM-систем

Система управління взаємовідносинами з клієнтами (CRM, CRM-система, Customer Relationship Management) - прикладне програмне забезпечення для організацій, призначене для автоматизації стратегій взаємодії з замовниками (клієнтами), зокрема для підвищення рівня продажів, оптимізації маркетингу і поліпшення обслуговування клієнтів шляхом збереження інформації про клієнтів і історії взаємин з ними, встановлення і поліпшення бізнес-процесів і подальшого аналізу результатів [1].

Насправді, CRM-системою можна вважати будь-який варіант контролю і обліку, який допоможе поліпшити взаємодію з клієнтами. Навіть якщо ведеться історія дзвінків і контактів на папері або в Excel - це теж можна вважати CRM-системою в тому випадку, якщо розроблена схема обліку і контролю працює і дозволяє контролювати всі варіанти взаємодії з клієнтами. Звичайно, такі методи ведення обліку відходять у минуле, адже в сучасному світі без ефективної автоматизації складно уявити собі роботу будь-якого бізнесу. А тому, коли говорять про CRM-системи, зазвичай мають на увазі спеціальне програмне забезпечення.

Іншими словами, CRM-система - це будь-яке програмне забезпечення, яке допомагає успішно контролювати роботу з клієнтами, упорядкувати і планувати її.

Необхідність впровадження CRM. CRM потрібна для того, щоб в першу чергу не втратити потенційного клієнта. У малому і середньому бізнесі нашої країни конкуренція дуже висока, тому компанії докладають значних зусиль для того, щоб залучити клієнтів. На такі типи витрат компанією виділяється значний бюджет і дуже важливо, щоб ці витрати виправдали себе.

По-друге, для контролю співробітників компанії і стандартизації роботи з клієнтами. Без загальної стандартизованої CRM-системи кожен співробітник буде працювати так, як він звик.

По-третє, для накопичення статистичної бази, що дуже важливо для розвитку будь-якого бізнесу. Завдяки використанню CRM-системи вся робоча інформація збирається в одній загальній базі в стандартизованому вигляді..

Класифікація CRM. Існують різні типи CRM-систем, які розрізняються рівнем обробки інформації і вирішуваних завдань. Наприклад, кредитно-фінансовій організації необхідно вести базу даних клієнтів, фіксувати в базі їх телефонні номери для зв'язку і реклами.

1. Операційна CRM. Головне завдання операційних CRM-систем - підвищити лояльність клієнта в процесі безпосереднього контакту з ним.

Такі системи спрощують взаємодію з клієнтами, систематизують дані про заявки і угодах, самі виставляють рахунки, нагадують передзвонити клієнтові і можуть самі відправити йому sms-повідомлення, записують телефонні дзвінки і т. д [2].

Всі операційні CRM вміють:

* Реєструвати вхідний трафік (дзвінки, листи, заявки на сайт);
* Зберігати в базі дані про клієнтів, заявках, угодах, завдання і т. д .;
* Автоматизувати документообіг всередині компанії;
* Фіксувати просування угод по воронці продажів;
* Нагадувати про заплановані дзвінках, листах, зустрічах;
* Ставити завдання і контролювати роботу співробітників.

2. Аналітична CRM. Мета аналітичних CRM - це аналіз накопиченої інформації про клієнтів і продажах для вибудовування більш ефективної стратегії.

На відміну від операційних систем, аналітичні не тільки фіксують історію взаємодії з клієнтом, але і допомагають простежити закономірності в продажах: клієнти з яких джерел купують найчастіше, на якому етапі зривається більшість угод, як розподілені клієнти по воронці продажів - і всі ці дані оновлюються в онлайн-режимі, в розрізі кожного параметра.

Аналітичні CRM вміють:

* Сегментувати клієнтську базу;
* Визначати цінність клієнта;
* Аналізувати їх рентабельність;
* Відстежувати поведінку клієнтів на кожному етапі операції;
* Показувати розподіл угод по воронці продажів;
* Аналізувати динаміку продажів;
* Аналізувати ефективність маркетингових інструментів;
* Прогнозувати обсяг продажів.

3. Колабораційна CRM. Такі CRM налагоджують комунікації з клієнтами для збору зворотного зв'язку. Інформація, отримана з їх допомогою, допомагає скорегувати асортимент товарів, цінову політику, а також процес обслуговування покупців. Наприклад, співробітники call-центру фінансово-кредитної організації обдзвонюють клієнтів, нагадуючи про платіжну дату або прострочену заборгованість, фіксуючи результати дзвінків в базі даних.

# 2.3. Особливості розробки та впровадження CRM систем

CRM відкриває широкі можливості для розвитку малого та великого бізнесу. Сучасним компаніям слід з великою обережністю та досить кваліфіковано керувати потенційними клієнтами та існуючими клієнтами. Використання платформи з найкращими функціями CRM є важливою частиною цієї стратегії.

Використання CRM-системи дозволяє отримати наступні переваги:

1. Впорядкування та автоматизація процесів: стандартизувати робочі процеси та процеси, пов’язані з маркетингом, підтримкою клієнтів та продажами, і таким чином покращити координацію.

2. Поліпшення обслуговування клієнтів: задоволені клієнти частіше стають лояльними клієнтами і частіше повертаються. Отже, надзвичайно важливо знати, чи задоволені всі клієнти. Якщо ні, то потрібно з’ясувати причини та виправити їх. Платформа CRM має важливе значення з точки зору збору даних про задоволеність споживачів. Деякі продукти також включають можливості обслуговування клієнтів для подальшої підтримки клієнтів.

3. Збільшення продажів: CRM дозволяє отримувати такі важливі дані клієнтів, як дні народження, місцезнаходження, вік та галузь, які можна використовувати для персоналізації спілкування, а також пропозицій. За допомогою правильних пропозицій, зроблених правильним клієнтам в оптимальний час, можна укладати вигідніші пропозиції, що призводить до поліпшення рентабельності інвестицій.

4. Вибір рішення. Правильний вибір CRM для організації має вирішальне значення для успіху клієнтів. По суті, це зводиться до двох ключових моментів. Щоб вибрати правильний CRM, потрібно: 1) ознайомитися з функціями CRM, щоб вибрати з них, і 2) знати, які конкретні можливості CRM потрібні команді.

5. Управління контактами. Управління контактами, елемент CRM, який іноді передається програмним забезпеченням для управління клієнтами, є життєво важливою можливістю CRM. Це дозволяє користувачам сегментувати контакти в групи для організації та кращого обслуговування. Завдяки цій CRM-можливості можна збирати важливі дані про клієнтів та зміцнювати свої маркетингові стратегії при впровадженні нових продуктів. Це також покращує персоналізацію всіх контактів із клієнтами, змушуючи їх почувати себе важливими та задоволеними.

6. Команда продажів та управління можливостями клієнтів. Кількісна оцінка можливостей команди продажів із клієнтами є відносно основною особливістю CRM.

7. Звіти та інформаційні панелі. Після переходу до інструментів бізнес-аналітики BI, звіти та інформаційні панелі стали одними з найкращих функцій CRM. Користувачі можуть переглядати статистику у візуальній перспективі за допомогою індивідуальних звітів та інформаційних панелей. Вони пропонують оновлення даних у реальному часі та платформу, до якої можна отримати доступ з різних порталів, покращуючи зручність доступу для різноманітного персоналу. Керівництво може використовувати ці звіти для прийняття рішень, керованих даними - інвестиція в CRM BI - це інвестиція в майбутнє бізнесу.

8. Аналітика продажів. Що стосується звітності, аналітика продажів є однією з найцінніших характеристик CRM. Користувачі можуть створювати кращі кампанії продажів у майбутньому, аналізуючи дані минулих кампаній. CRM допомагає збирати дані із соціальних мереж, опитувань та відвідуваності веб-сайтів, а потім аналізувати їх - все за допомогою програмного забезпечення.

9. Мобільний CRM. У наш час у кожного є смартфон, тому має сенс використовувати технології всередині них, щоб принести користь бізнесу. Тому доцільно керувати відносинами з клієнтами на віддаленому пристрої за допомогою мобільних програм CRM та програм. Персонал може переглядати дані та отримувати цінні сповіщення про кампанії на мобільному пристрої.

10. Автоматизація продажів. Ще однією з багатьох функцій CRM є можливість автоматизувати певні аспекти процесу продажу. Це включає вихідні дзвінки та подальші кампанії, а також організацію даних для більш ефективних майбутніх кампаній. Автоматизація цих повторюваних завдань дозволяє працівникам приділяти свій час та зосереджуватися на більш складних завданнях.

11. Прогнозування продажів. Прогнозування дозволяє створити орієнтир, за допомогою якого можна визначити, чи насправді результати відповідають зусиллям організації. Також можна використовувати ці дані, щоб визначити, куди потрібно спрямувати подальші зусилля.

12. Інтеграція клієнтів електронної пошти. Також можна отримати всю автоматизацію та організацію спеціального поштового клієнта всередині інтерфейсу користувача CRM. Тоді, коли клієнт телефонує, що потребує підтримки, представники можуть легко отримати інформацію про попередні взаємодії, щоб задовольнити вимоги клієнта вчасно та якісно.

13. Оптимізація робочих процесів. Одним із способів, за допомогою якого компанія суттєво підвищить свою ефективність, є оптимізація робочих процесів. CRM можуть полегшити оптимізацію шляхом автоматизації таких процесів, як збір даних, аналіз даних, маркетингові кампанії та інші завдання, які раніше виконувались вручну. Вони також пропонують звіти та аналітику, щоб допомогти користувачам визначити проблемні області, щоб покращити їх.

14. Дані про продажі. Однією з найкращих особливостей CRM є здатність організувати продажі та дані про клієнтів у керовані частки. Після того, як необроблені дані стратифікуються, їх можна перетворити та візуалізувати, що зручно для прийняття рішень. Користувачі можуть кількісно визначати KPI, визначати больові точки та приймати рішення, керовані даними.

15. Зберігання даних/файлів CRM. Щоб максимізувати корисність даних, їх потрібно правильно зберігати. Однією з ключових особливостей CRM є запатентований спосіб зберігання даних, завдяки якому користувачі можуть ефективно збирати дані. CRM також пропонують захисні засоби для захисту та резервного копіювання цих важливих даних, щоб запобігти втратам та порушенням безпеки.

16. Синхронізація та обмін файлами. CRM функціонує як платформа синхронізації для багатьох зовнішніх програм. Можна завантажувати електронні листи з Outlook, синхронізувати з Google Drive, імпортувати електронні таблиці тощо. Через можливості спільного використання CRM можна переглядати весь набір функцій на одній платформі.

17. Управління ефективністю продажів. Серед інших функцій CRM, управління ефективністю продажів пропонує ряд переваг для команд продажів. Менеджери можуть легко побачити, які аспекти команди продажів працюють добре, а які потребують вдосконалення. Дані можуть бути організовані торговим представником або іншими суб’єктами. Можна керувати ефективністю роботи партнерів з продажу на платформі, щоб переконатися, що команда працює достатньо ефективно.

18. Інтеграція автоматизації маркетингу. Автоматизація маркетингу дозволяє більш ефективно та стратегічно націлювати аудиторію, яка, ймовірно, буде зацікавлена ​​у товарах чи послугах організації.

19. Інтеграція чату. Функція чату є актуальною для сучасного користувача. Інтегруючи програму чату, як Olark або LivePerson, безпосередньо з CRM, можна спілкуватися в чаті зі співробітниками, партнерами та надавати підтримку клієнтам з одного інтерфейсу користувача.

20. Автоматизація та інтеграція кол-центру. Однією з ключових особливостей CRM є можливість керувати своїм кол-центром за допомогою інтерфейсу CRM та інтегрувати будь-яку із сторонніх програм в єдину платформу. Чим менше різних систем використовується, тим менша ймовірність заплутатися та втратити інформацію.

21. Інтеграція веб-аналітики. Інтеграція веб-аналітики не просто дозволяє збирати дані про те, хто відвідує веб-сайт. Вона аналізує необроблені дані, визначає проблемні області, пропонує шляхи вдосконалення та надає уявлення про взаємодію бази користувачів із вашим веб-сайтом. CRM-системи забезпечують інтуїтивні способи збору та доступу до цієї статистики з інтерфейсу користувача.

22. Автоматизація підтримки. Функції підтримки клієнтів, такі як сторінки поширених запитань, чат-боти та листування електронною поштою, стають дедалі автоматизованішими, оскільки можливості систем покращуються. Користувачі можуть контролювати вихід функцій підтримки клієнтів з інтерфейсу.

23. Тестування середовища. Ця функціональність CRM стосується здатності користувачів тестувати нові можливості перед їх розповсюдженням. Клієнти можуть досліджувати та надавати відгуки, а розробник збирає дані про кожну взаємодію.

24. Інтеграція управління соціальними медіа. Спеціальними зусиллями SMM можна керувати за допомогою інтерфейсу CRM. Можна інтегрувати платформи управління, такі як Oktopost та Hootsuite, щоб впорядкувати управління соціальними мережами та дозволити кільком членам команди працювати разом із різних точок доступу.

# Висновки за розділом 2

Більшість CRM-систем малого бізнесу не включають багато способів складання звітів, якщо такі є. Найкращі системи включатимуть розробника звітів з прогнозу продажів, а також розробника звітів для спеціальної діяльності.

На сучасному ринку, навіть для малого бізнесу, автоматизація CRM стає необхідною. Автоматизація допомагає підвищити ефективність та продуктивність за рахунок автоматизації ручних процесів. Різні дії в CRM запускають робочі процеси для виконання завдання відповідно до набору правил, який визначаються. Це може означати активацію процесу виставлення рахунків при закритті продажу або нагадування електронною поштою про закінчення терміну гарантії.

CRM із інтегрованою системою довідкової служби означає, що також можна контролювати взаємодію клієнтів після продажу. Будь-які підняті питання підтримки будуть враховані, тому команда продажів може бути попереджена про будь-які потенційні проблеми.

Організації, що ведуть бізнес до бізнесу (B2B), як правило, є цільовою аудиторією для постачальників CRM, але програмне забезпечення також може мати значення для інших ринків.

Зазвичай CRM-система, яка найчастіше використовується спеціалістами з продажу та маркетингу, допомагає організаціям впорядкувати свій робочий процес, використовуючи автоматизацію продажів (SFA), покращуючи ефективність та продуктивність, стимулюючи ріст бізнесу та збільшуючи задоволеність клієнтів.

Система CRM, що використовується в основному для управління контактами та управління продажами, забезпечує єдиний центр, який зберігає всі дані для ділових контактів. Це може включати клієнтів, потенційних клієнтів, постачальників, партнерів, пресу та колег.

# РОЗДІЛ 3. РОЗРОБКА ВЕБ-ПРОЕКТУ

# 3.1. Постановка завдання

Потрібно розробити CRM-систему для фінансово-кредитної організації і особистий кабінет за допомогою веб-фреймворку Django, де CRM-менеджери можуть додавати нові кредити, відстежувати клієнтів і вести їх кредитні справи.

Програмне забезпечення повинне забезпечувати наступні функціональні можливості:

1. Додавання даних в базу даних;
2. Зчитування даних з бази даних;
3. Стилізація шаблонів за допомогою CSS-фреймворка Bootstrap;
4. Динамічну зміну кількості кредитів та клієнтів, а також динамічне видалення даних;
5. Відображення та керування даними в адміністративній панелі.

# 3.2. Вибір інструментів для реалізації

На сьогоднішній день існує безліч способів реалізації CRM-системи за допомогою веб-фреймворку, наприклад, AllcountJS, Zend, Yii, Django і т. д. Для реалізації свого проекту я вибрав саме веб-фреймворк Django. Django це опенсорсний фреймворк для веб-додатків написаний на мові Python.

Python являє собою популярну, високорівневу мову програмування, що призначена для створення додатків різних типів. Це і веб-додатки, і ігри, і десктоп-програми, і робота з базами даних. Досить велике поширення Python отримав в області машинного навчання і досліджень штучного інтелекту.

Спочатку Django був створений в 2005 році для управління новинними сайтами LJWorld.com, lawrence.com і KUsports.com, що належать компанії The World Company. У 2008 році вийшов публічний перший реліз фреймворку. На сьогоднішній день він продовжує розвиватися. Поточною версією фреймворку є версія 3.2.3 вийшла 13 травня 2021 року. Також постійно виходять підверсії, які доповнюють фреймворк і виправляють нечисленні помилки [4].

Django досить популярний. Він використовується на багатьох сайтах, таких як Instagram, Disqus, Mozilla, The Washington Times, Pinterest, YouTube, Google та ін.

Також Django використовується в якості веб-компонента в різних проектах, таких як Graphite - система побудови графіків і спостереження, FreeNAS - вільна реалізація системи зберігання та обміну файлами і ін.

Фреймворк Django реалізує архітектурний патерн Model-View-Template або скорочено MVT, який за фактом є модифікацією поширеного в веб-програмуванні патерну MVC (Model-ViewController, Модель-Подання-Контролер).

Схематично можна уявити архітектуру MVT в Django наступним чином (Рис. 2):

**Browser**

**Template**

**URL dispatcher**

**View**

**Model**

**DataBase**

Рис. 3.1. Архітектура MVT

Основні елементи патерну:

* URL dispatcher: при отриманні запиту на підставі запитаної адреси URL визначає, який ресурс повинен обробляти даний запит;
* View: отримує запит, обробляє його і відправляє у відповідь користувачеві певну відповідь. Якщо для обробки запиту необхідно звернення до моделі і бази даних, то View взаємодіє з ними. Для створення відповіді може застосовувати Template або шаблони. В архітектурі MVC цьому компоненту відповідають контролери (але не представлення);
* Model: описує дані, які використовуються в додатку. Окремі класи, як правило, відповідають таблицями в базі даних;
* Template: представляє логіку представлення у вигляді згенерованої розмітки html. У MVC цьому компоненту відповідає View, тобто представлення.

Коли до додатку приходить запит, то URL dispatcher визначає, з яким ресурсом зіставляється даний запит і передає цей запит обраному ресурсу. Ресурс фактично являє функцію або View, який отримує запит і певним чином обробляє його.

В процесі обробки View може звертатися до моделей і бази даних, отримувати з неї дані, або, навпаки, зберігати в неї дані. Результат обробки запиту відправляється назад, і цей результат користувач бачить в своєму браузері. Як правило, результат обробки запиту представляє згенерований html-код, для генерації якого застосовуються шаблони (Template).

З плюсів даного веб-фреймворку можна виділити:

* Швидкість: Django був розроблений, щоб допомогти розробникам створити додаток швидко настільки, на скільки це можливо. Це включає в себе формування ідеї, розробку і випуск проекту, де Django економить час і ресурси на кожному з цих етапів. Таким чином, його можна назвати ідеальним рішенням для розробників, для яких питання дедлайну варто в пріоритеті;
* Повна комплектація: Django працює з десятками додаткових функцій, які суттєво допомагають з аутентифікацією користувача, картами сайту, адмініструванням вмісту і багатьом іншим. Дані аспекти допомагають здійснити кожен етап веб-розробки;
* Безпека: Працюючи в Django, ви отримуєте захист від помилок, пов'язаних з безпекою які ставлять під загрозу проект. Я маю на увазі такі поширені помилки, як ін'єкції SQL, крос-сайт підробки, clickjacking і крос-сайтовий скриптинг. Для ефективного використання логінів і паролів, система користувальницької аутентифікації є ключем;
* Масштабованість: фреймворк Django найкращим чином підходить для роботи з найвищими трафіками. Отже, логічно, що велика кількість завантажених сайтів використовують Django для задоволення вимог, пов'язаних з трафіком;
* Різнобічність: менеджмент контенту, наукові обчислювальні платформи, навіть великі організації - з усім цим можна ефективно справлятися за допомогою Django.

Також варто відзначити, що для роботи з даним веб-фреймворком немає необхідності встановлювати IDE редактори. Досить встановити інтерпретатор Python і встановити в нього Django.

Для того щоб встановити Django, потрібен пакетний менеджер pip. pip - система управління пакетами, яка використовується для установки і управління програмними пакетами, написаними на Python.

Для стилізації шаблонів в проекті мною було використано CSS фреймворк Bootstrap. Bootstrap - це CSS фреймворк, який спочатку створювався для внутрішнього використання компанією «Twitter» з робочою назвою «Twitter Blueprint», але в підсумку був опублікований у відкритий доступ і став хорошим набором інструментів для front-end розробки під назвою «Bootstrap» [8].

Переваги Bootstrap:

* Висока швидкість розробки макетів сторінок сайту. Bootstrap містить величезний набір готових рішень і елементів;
* Кросбраузерність і адаптивність сайту. Всі елементи фреймворка адаптивні під всі пристрої і коректно відображаються у всіх сучасних браузерах;
* Легкість у використанні. Навіть людина, що має базові знання про HTML і CSS, може вільно створювати web-сторінки з використанням фреймворку;
* Простота в навчанні. У Bootstrap дуже хороша документація з великою кількістю прикладів готового коду.

Основні інструменти Bootstrap:

* Сітки - заздалегідь задані розміри колонок, які можна відразу ж використовувати, наприклад, ширина колонки 140 px відноситься до класу .span2 (.col-md-2 в третій версії фреймворка), який можна використовувати в CSS описі документа;
* Шаблони - фіксований або еластичний шаблон документа;
* Типографіка - опису шрифтів, визначення деяких класів для шрифтів, таких як код, цитати і т. д .;
* Медіа - надає деяке упорядкування зображень та відео.
* Таблиці - засоби оформлення таблиць, аж до додавання функціональності сортування;
* Форми - класи для оформлення форм і деяких подій, що відбуваються з ними;
* Навігація - класи оформлення для панелей, вкладок, переходу по сторінках, меню і панелі інструментів;
* Алерт - оформлення діалогових вікон, підказок і спливаючих вікон.

Для зберігання даних як правило необхідна база даних.

База даних - сукупність даних, організованих відповідно до концептуальної структурою, яка описує характеристики цих даних і взаємини між ними.

Існує величезна кількість різновидів баз даних, що відрізняються за різними критеріями.

Класифікація за моделлю даних:

* ієрархічна;
* Об'єктна і об'єктно-орієнтована;
* Об'єктно-реляційна;
* реляційна;
* мережева;
* Функціональна.

Django проектувався для роботи під управлінням Apache з модулем mod python і з використанням PostgreSQL в якості бази даних.

В даний час, крім бази даних PostgreSQL, Django може працювати з іншими СУБД: MySQL, SQLite і Oracle.

У даній роботі використовується база даних SQLite. SQLite - це база даних, чимось схожа на MySQL. Принципова відмінність SQLite від інших БД в тому, що вся база являє собою один файл. Якщо в MySQL база зберігається десь в нетрях сервера і недоступна для перенесення, то в SQLite просто: один файл - одна база.

# 3.3 Програмна реалізація веб-проекту

**3.3.1 Опис концепції розроблюваного веб-проекту**

Концепція розроблюваного програмного продукту являє собою веб-проект, побудований на веб-фреймворку Django, що надає наступні можливості:

* Відображення загальної інформації про кількість кредитів на головній сторінці;
* Відображення списку клієнтів на головній сторінці;
* Відображення списку кредитів на головній сторінці;
* Відображення інформації про клієнта та його кредити на окремій сторінці;
* Відображення списку кредитних продуктів на окремій сторінці;
* Додавання нових клієнтів в адміністративній частині додатку CRM-менеджерами;
* Додавання нових кредитів в адміністративній частині додатку CRM-менеджерами;

Так як веб-проект служить в першу чергу як CRM-система, в якості критеріїв ефективності виступлять основні функції, які повинні бути присутніми в CRM-системі. У список основних функцій CRM-системи мого веб-проекту входять:

* Відстеження клієнтів;
* Відстеження кредитів.

**3.2.2 Архітектура веб-проекту**

Проект складається з двох частин:

* Загальнодоступна частина, яка по суті відображає інформацію з БД в зручному вигляді, і дозволяє CRM-менеджерам відстежувати клієнтів;
* Адміністративна частина, яка дозволяє CRM-менеджерам заповнювати і редагувати базу даних сайту.

Ці дві частини реалізовані, відповідно до шаблону MVT, описаного вище. MVT визначає спосіб розробки програмного забезпечення, за якого код для визначення і доступу до даних (модель, файл models.py) відділений від логіки додатку (управління, файл views.py), яка в свою чергу відокремлена від інтерфейсу користувача (представлення, файл html).

При створенні веб-проекту на Django відбувається поділ на власне веб-проект і веб-додаток, який створюється після створення проекту.

**3.3.3 Структура веб-проекту**

В основу проекту входять:

* Зовнішній каталог d\_app/ - це просто контейнер для проекту. Його назва не використовується Django, і його можна перейменувати в що завгодно;
* Внутрішній каталог crm1/ - це пакет Python проекту. Його назва - це назва пакету Python, яка буде використовуватися для імпорту будь-чого з проекту.

У внутрішньому знаходяться каталоги:

1. Каталог accounts — це веб-додаток проекту;
2. Каталог static — каталог, в якому знаходяться статичні файли та різні медіа файли що використовуються в проекті.
* База даних db.sqlite3 котра знаходиться в каталозі crm1, використовується в проекті. В Django база даних створюється після того як виконається перша міграція в веб-додатку.
* Файл manage.py, котрий знаходиться в каталозі crm1 – це скріпт, який дозволяє взаємодіяти з проектом Django.

Також у внутрішньому каталозі crm1 ствоюється каталог з однойменною назвою, який містить файли:

* Файл settings.py – це файл, який містить конфігурації проекту;

Розглянемо деякі конфігурації:

1. DATABASES - словник містить налаштування для всіх баз даних,

які будуть використовуватися Django. Словник містить псевдоніми використовуваних баз даних і словник з налаштуваннями для кожної. Налаштування DATABASES повинне визначати базу даних з псевдонімом default і будь-яку кількість додаткових баз даних. У даній роботі представлений найпростіший варіант – це одна SQLite база даних.

ENGINE - бекенд бази даних (так як використовувалася СУБД SQLite, то був узятий бекенд django.db.backends.sqlite3).

NAME – назва бази даних. Значення по замовчуванню, os.path.join (BASE\_DIR, 'db.sqlite3'), створить файл в каталозі проекту;

2. INSTALLED\_APPS – Список строк, який вказує на всі додатки Django, використовувані в проекті.

За замовчуванням, INTALLED\_APPS містить наступні додатки, які надаються Django:

1. django.contrib.admin - інтерфейс адміністратора;
2. django.contrib.auth - система авторизації;
3. django.contrib.contenttypes - фреймворк типів даних;
4. django.contrib.sessions - фреймворк сесії;
5. django.contrib.messages - фреймворк повідомлень;
6. django.contrib.staticfiles - фреймворк для роботи зі статичними файлами.

Також було додано додаток accounts – власне додаток який ми створили;

3. STATIC\_URL – шлях до каталогу, який містить всі статичні файли;

4. MEDIA\_URL – шлях до каталогу, який містить всі статичні файли;

* Файл urls.py, що знаходиться в каталозі crm1 – конфігурація URL-ів проекту Django. Це «зміст» усіх Django-сайтів. Для визначення URL-ів додаткe, використовується модуль Python неофіційно названий URLconf (конфігурація URL-ів). Цей модуль містить код Python, який відображає URL-шаблони (регулярні вирази) і пов'язані функції Python.

Для додавання URL-ів з модуля django.urls беруться атрибути url і include;

* Файл wsgi.py, що знаходиться в каталозі DjangoTravel - це точка входу проекту для WSGI - спільних веб-серверів.

**3.3.4 Структура веб-додатку**

В веб-додаток входять:

* Каталог accounts - це каталог, в якому знаходяться файли і каталоги додатка. В каталог accounts входить каталог migrations. Це каталог, який містить всі файли міграцій. Django створить міграції при будь-яких змінах моделі або полів - навіть тих параметрів, які не впливають на базу даних - так як єдиний спосіб відновити стан моделей - це зберігати всі зміни в історії. «Початкові міграції» програми – це міграції, які створюють першу версію таблиць для додатка. Зазвичай додаток містить тільки одну початкову міграцію, але для складної структури моделей їх може бути кілька. «Початкові міграції» позначаються атрибутом initial = True в класі міграції. Якщо атрибут initial не вказано, міграція буде вважатися «початкової», якщо це перша міграція в додатку (тобто вона не залежить від іншої міграції поточного додатка).

Також в каталог accounts входить каталог templates в якому зберігаються шаблони веб-додатку;

* models.py - файл відповідає за моделі в Django. Моделі відображають інформацію про дані. Вони містять поля і поведінку даних. Зазвичай одна модель представляє одну таблицю в базі даних.

В даній роботі було створено три моделі з назвами Customer, Product, Order.

Кожна модель представлена класом, успадкованим від django.db.models.Model. Кожна модель містить певну кількість атрибутів, кожний з яких відображає поле в табиці бази даних.

Кожне поле представлене екземпляром класу Field – наприклад, CharField для текстових полів. Це вказує Django які типи даних зберігають ці поля.

Назви кожного екземпляру Field – це назви поля, в «машинному» (machine-friendly) форматі. Ці назви будуть використовуватись в коді, а база даних буде використовувати їх як імена полів.

Деякі класи, успадковані від Field, мають обов’язкові аргументи. Наприклад, CharField вимагає, щоб йому передавали max\_length. Це використовується не лише в семі бази даних, а і при валідації. Також в моделях присутній необов’язковий аргумент verbouse\_name = “ ” – це читабельна назва моделі, в одинині. Якщо цей параметр не вказаний, то в адмінітративній частині відобразиться ім’я поля. Додаткові налаштування для моделі можна визначити через class Meta, наприклад назву в одинині чи множині (verbose\_name і verbose\_name\_plural). Всі вони не обов’язкові, і додавати class Meta також не обов’язково.

* admin.py – файл необхідний для використання моделей в адміністративній частині проекту. Код файлу вказаний на рисунку нижче:



Рис. 3.2. Код файлу admin.py

В файл імпортуються всі використовувані моделі і з допомогою команди admin.site.register реєструються для відображення в адміністративній частині;

* views.py – файл який відповідає за відображення загальнодоступної частини. По суті це логіка додатку. Відображення є функцією, яка обробляє HTTP-запит, отримую дані із ази даних, які застосовуються для генерації сторінки HTML. Потым функція відображення повертає згенеровану сторінку користувачу в вигляді HTTP-відповіді (Додаток А);
* urls.py – файл, котрий відповідає за зв’язок з файлом urls.py, котрий знаходиться в основі проекту. Функція url() визначає URL-паттерн і функцію відображення, яка буде викликана, якщо введена адреса буде відповідати даному паттерну.

Дана функція url(), окрім того, визначає параметр name, який унікально визначає це одиничне URL-перетворення.

* apps.py – файл, який визначає конфігурацію додатку. Код файлу apps.py представлений на рисунку нижче:



Рис. 3.3. Код файлу apps.py

Django містить реєстр встановлених додатків, який містить поточні налаштування і надає інтроспекцію. Також надає лсит доступних моделей.

Цей реєстр називається просто apps і знаходиться в модулі django.apps. Повна схема роботи веб-додатку представлена на рисунку 3.4.



Рис. 3.4. Схема роботи веб-додатку

# 3.4 Опис функціональності

**3.4.1 Загальнодоступна частина**

При вході на сайт кориистувач бачить зовнішню(користувацьку) частину веб-додатку.(Рис. 3.5)



Рис. 3.5. Головна сторінка «Dashboard»

Головна сторінка поділена на декілька блоків. У верхньому блоці знаходиться панель навігації, яка дозволяє переходити між сторінками Dashboard та Products.

Нижче знаходиться панель, яка показує загальну кількість заявок відповідно до їх статусу: «Всього заявок», «Кількість відхилених заявок», «Кількість схвалених активних заявок», «Кількість заявок, які очікують на розгляд». Ця інформація зберігається в базі даних, і відповідно може бути змінена менеджером, який ці статуси вказує (Рис. 3.6).



Рис. 3.6. Статуси заявок.

Третій блок поділений на дві частини. Зліва знаходиться таблиця, яка відображає клієнтів що є в системі (Рис. 3.7).



Рис. 3.7. Блок «Користувачі»

Ця таблиця містить кнопку «View», яка дозволяє перейти на картку відповідного клієнта (Рис. 3.8).



Рис. 3.8. Картка клієнта

Дана сторінка веб-додатку містить інформацію про клієнта: ПІБ, контакти, список та загальну кількість його заявок, а також їх поточні статуси.

Повернемось на вкладку «Dashboard», та розглянемо таблицю «Orders» (Рис. 3.9).



Рис. 3.9. Блок «Orders»

Ця таблиця відображає список усіх заявок, їх статус, ким та коли вони були зроблені. Також, таблиця містить поле «Product». Це код кредитного продукту, детальна інформація про який зберігається на вкладці «Products», в верхній панелі навігації (Рис. 3.9).



Рис. 3.10. Сторінка «Products»

**3.4.2 Адмінастативна частина**

Для входу в адмін-панель потрібно перейти на сторінку адміністратора. Для цього, в адресній строці дописати «/admin», після чого завантажиться форма для входу в адміністративну частину сайту (Рис. 3.11). Тут потрібно ввести ім’я користувача та пароль у відповідні поля. Необхідні «username» та «password» можна створити за допомогою команди python manage.py createsuperuser.



Рис. 3.11. Форма для входу в адміністративну панель

Якщо ввести некоректні дані, то з’явиться повідомлення про помилку Рис. 3.12).



Рис. 3.12. Сповіщення про помилку входу в адміністративну частину

Після того як були введені коректні дані необхідно натиснути на кнопку «Log in». Після цього CRM-менеджер побачить стартовий інтерфейс Адміністрування Django (Рис. 3.13).



Рис. 3.13. Стартовий інтерфейс адміністративної частини

В цій частині сайту відображаються всі моделі(таблиці) згруповані по встановлених додатках. CRM-менеджер має можливість клікнути по назві моделі, щоб отримати список всіх пов’язаних записів, далі можна клікнути по цих записах, для їх редагування. Також можна безпосередньої клікнути на кнопку «Add+», розміщену поруч з кожною моделлю щоб почати створення нового запису цього типу. Додати новий запис можна натиснувши на кнопку «Add+», що знаходиться на кожній із послідуючих сторінок.

Для перегляду клієнтів CRM-менеджер має клікнути на назву моделі «Customers». Після цього він перейде на сторінку де будуть відображені клікабельні посилання з клієнтами (Рис. 3.14):



Рис. 3.14. Сторінка «Customers»

Якщо клікнути по користувачу, то відобразиться сторінка із формами вводу, де інформацію по користувачу можна редактувати, а бо ж взагалі видалити (Рис. 3.15):



 Рис. 3.15. Сторінка «Change customer»

Клікнувши по моделі «Orders», CRM-менеджер зможе побачити усі кредитні заявки (Рис. 3.16):



Рис. 3.16. Сторінка «Orders»

І аналогічно, як і в попередньому випадку, натиснувши на клікабельне посилання відобразиться інтерфейс де можна редактувати інформацію даної моделі (Рис. 3.17):



Рис. 3.17. Сторінка «Change order»

Тут можна видалити запис, зберегти запис і продовжити редактування і додати новий запис. При збереженні, в відповідній таблиці бази даних зміняться збережувані дані. Запис в базу даних проходить за тим же самим принципом.

Деякі поля можна заповнити лише значеннями із випадаючого списку, так як між таблицями прописані зв’язки. Наприклад в полі customer можна вибрати лише користувача який вже є в базі даних системи (Рис. 3.18).



Рис. 3.18 Редагування запису в таблиці «Orders»

Якщо ж потрібна інформація по якихось причинах відсутня в базі даних, її можна легко додати без складних додаткових маніпуляцій. Достатньо натиснути «+» поряд із потрібною формою. У випадку поля customer з’явиться нове вікно із формою додавання запису в таблицю «Customers» (Рис. 3.19):



Рис. 3.19. Форма додавання запису «Add customer»

По аналогії з вищеописаним такі ж самі маніпуляції можна провести з моделлю «Products», де міститься інформація про кредитні продукти (Рис. 3.20).



Рис. 3.20. Сторінка «Change products»

Варто відзначити, що в цьому випадку, в полі Category, нема безпосередньої можливості додати записи в випадаючий список, так як вони не пов’язані з іншими таблицями в БД, а прописані при створенні самої моделі.(Додаток Б)

# Висновки за розділом 3

На сьогоднішній день існує безліч способів реалізації CRM-системи за допомогою веб-фреймворку, наприклад, AllcountJS, Zend, Yii, Django і т. д. Для реалізації свого проекту я вибрав саме веб-фреймворк Django. Django це опенсорсний фреймворк для веб-додатків написаний на мові Python.

Для стилізації шаблонів в проекті мною було використано CSS фреймворк Bootstrap. Bootstrap - це CSS фреймворк, який спочатку створювався для внутрішнього використання компанією «Twitter» з робочою назвою «Twitter Blueprint», але в підсумку був опублікований у відкритий доступ і став хорошим набором інструментів для front-end розробки під назвою «Bootstrap» [8] .

У даній роботі використовується база даних SQLite. SQLite - це база даних, чимось схожа на MySQL. Принципова відмінність SQLite від інших БД в тому, що вся база являє собою один файл. Якщо в MySQL база зберігається десь в нетрях сервера і недоступна для перенесення, то в SQLite просто: один файл - одна база.

Так як веб-проект служить в першу чергу як CRM-система, в якості критеріїв ефективності виступлять основні функції, які повинні бути присутніми в CRM-системі. У список основних функцій CRM-системи розробленого веб-проекту входять:

* Відстеження клієнтів;
* Відстеження кредитів.

# ВИСНОВКИ

Впровадження CRM-системи є одним з пріоритетних напрямів розвитку будь-якої компанії, оскільки може забезпечити підвищення якості обслуговування клієнтів, зменшити затрати праці на супроводжування і звільнити працівників від рутинної роботи.

CRM-системи автоматизують процеси взаємовідносин з клієнтами, тим самим допомагають впроваджувати, вимірювати і контролювати внутрішні стандарти і методики роботи з клієнтами.

Кожна CRM-система – це втіленні бачення розробників того, як потрібно працювати з клієнтом. В ній закладено багато готових інструментів, які дозволяють перевести роботу на якісно новий рівень.

Керівник не має на що опиратись в даному питанні, а тому і відділ роботи з клієнтами часто працює не найкращим чином. Впровадження CRM-системи дозволяє отримати не лише інструмент, але і допомогу, погляд розробників на те, як повинен працювати відділ. В свою чергу при розробці CRM-системи зазвичай опираються на найкращі практики, на експертів в питаннях роботи з клієнтами.

Метою написання даної бакалаврської роботи була розробка CRM-системи кредитно-фінансової організації засобами фреймворка Django.

В ході розробки була досліджена предметна область, проаналізовані вимоги до веб-додатку. Був спроектований і розроблений сайт з адміністративною панеллю, а також була спроектована база даних. Планується подальший розвиток проекту, зокрема, розробка особистого кабінету клієнта та форми реєстрації, допрацювання користувацького інтерфейсу. Все це можливе завдяки потужностям фреймворку Django. Як наслідок, додаток не втратить актуальності і лише додасть в корисності.

В ході розробки мною було досліджено та проаналізовано великий об’єм документації по технологіях фреймворку Django. Отримані навики розробки веб-додатків за допомогою даного фреймворку і верстки веб-сторінок з використанням сучасних технологій.

Для стилізації шаблонів в проекті мною було використано CSS фреймворк Bootstrap. Bootstrap - це CSS фреймворк, який спочатку створювався для внутрішнього використання компанією «Twitter» з робочою назвою «Twitter Blueprint», але в підсумку був опублікований у відкритий доступ і став хорошим набором інструментів для front-end розробки під назвою «Bootstrap» [8] .

У даній роботі використовується база даних SQLite. SQLite - це база даних, чимось схожа на MySQL. Принципова відмінність SQLite від інших БД в тому, що вся база являє собою один файл. Якщо в MySQL база зберігається десь в нетрях сервера і недоступна для перенесення, то в SQLite просто: один файл - одна база.

Так як веб-проект служить в першу чергу як CRM-система, в якості критеріїв ефективності виступлять основні функції, які повинні бути присутніми в CRM-системі. У список основних функцій CRM-системи розробленого веб-проекту входять:

* Відстеження клієнтів;
* Відстеження кредитів.

# СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Гринберг П. CRM со скоростью света / П. Гринберг – СПб.: Символ Плюс, 2016. – 530 с.
2. Кривошеева Р. Что такое CRM-системы и как их правильно выбирать? / Хабр [Електронний ресурс] / Р. Кривошеева // Habrahabr: Habrahabr.ru. – Режим доступу: https://habr.com/post/249633/.
3. Головатый А. Django. Подробное руководство /А. Головатый, Дж. Каплан-Мосс – СПб.: Символ-Плюс, 2010. – 560 с
4. Чан У. Django. Разработка веб-приложений на Python/ У. Чан, П. Биссекс, Д. Форсье – СПб.: Символ-Плюс, 2016. – 456 с.
5. Фримен А. jQuery для профессионалов/ А. Фримен – М.: «Вильямс», 2013. — 960 с.
6. Шевчук А. jQuery для начинающих / Хабр [Електронний ресурс] / А. Шевчук // Habrahabr: Habrahabr.ru.– Режим доступу: <https://habr.com/post/38208/>.
7. Bootstrap 3. Документація українською [Електронний ресурс]: документація для Bootstrap 3. – Режим доступу: <https://twbs.docs.org.ua/getting-started/>.
8. Марданов А., Мунасипов Р. Повышение эффективности клиентоориентированного бизнеса на основе интеграции технологий контакт-центра и CRM систем. URL: [www.intalev.ru](http://www.intalev.ru)
9. Албитов А., Соломатин Е. CRM (Customer Relationship Management). Корпоративный менеджмент. URL: <http://www.cfin.ru/itm/crm-review.shtml>
10. Ганущак-Єфіменко Л.М. CRM-система як ефективний інструмент розвитку готельного бізнесу в Україні. Вісник Київського національного університету технологій та дизайну. Серія : Економічні науки. 2017. № 4. С. 51–56.
11. Чайковська М.П. Перспективи гипермедійної інтеграції CRM-систем. Экономический вестник университета. Сборник научных трудов ученых и аспирантов. 2012. № 18-2. С. 201–207.
12. Ліщинська Л.Б. Основні аспекти автоматизації роботи з клієнтами засобами CRM-систем. Вісник Хмельницького національного університету. Економічні науки. 2015. № 5(1). С. 206–209.
13. Пейн Э. Руководство по CRM. Путь к совершенствованию менеджмента клиентов. М.: Вильямс, 2002. 824 с.
14. Функции CRM-систем. Независимый CRM-портал. URL: <http://www.crmonline.ru/>
15. Типы CRM-систем. Компания SalesapCRM. URL:https://salesap.ru/vidy-crm-sistem/
16. CRM-системи стали найбільшим сегментом ринку в 2017 році. URL: <https://news.finance.ua>
17. Результати дослідження ринку CRM в Україні. URL: <https://www.bitrix24.ua>
18. Сьомкіна Т.В., Литвинова О.В., Лобань О.О. Особливості моделей функціонування IT-компаній в Україні. Науковий вісник Ужгородського національного університету. Серія : Міжнародні економічні відносини та світове господарство. 2018. Вип. 19(3). С. 84–87
19. Рейтинг и Каталог CRM-систем популярных в Украине. URL: <https://срм.укр>
20. Бутенко Н.В. Впровадження концепції CRM на промисловому ринку. Економіка та держава. 2011. № 3. С. 40–42.
21. Ушакова І.О. Соціальні мережі, як засіб впливу на взаємовідносини з клієнтами. Системи обробки інформації. 2012. Вип. 8. С. 54–58.
22. Марданов А., Мунасіпов Р. Підвищення ефективності клієнтського бізнесу на основі інтеграції технологій контакт-центру і CRM систем. URL: www.intalev.ru (дата Звернення 03.01.2019).
23. Албітов А., Соломатін Е. CRM (Customer Relationship Management). Корпоративний менеджмент. URL: http://www.cfin.ru/itm/crm-review.shtml (дата Звернення 04.01.2019).
24. Ганущак-Єфіменко Л.М. CRM-система як Ефективний інструмент розвитку готельного бізнесу в Україні. Вісник Київського національного університету технологій та дизайну. Серія: Економічні науки. 2017. № 4. С. 51-56.
25. Чайковська М.П. Перспективи гіпермедійної інтеграції CRM-систем. Економічний вісник університету. Збірник наукових праць вчених і аспірантів. 2012. № 18-2. С. 201-207.
26. Ліщинська Л.Б. Основні аспекти автоматизації роботи з клієнтами засоби CRM-систем. Вісник Хмельницького національного університету. Економічні науки. 2015. № 5 (1). С. 206-209.
27. Пейн Е. Керівництво по CRM. Шлях до вдосконалення менеджменту клієнтів. М .: Вільямс, 2002. 824 с.
28. Функції CRM-систем. Незалежний CRM-портал. URL: http://www.crmonline.ru/ (дата Звернення 08.01.2019).
29. Типи CRM-систем. Компанія SalesapCRM. URL: https://salesap.ru/vidy-crm-sistem/ (дата Звернення 08.01.2019).
30. CRM-системи стали найбільшім сегментом Сайти Вся в 2017 году. URL: https://news.finance.ua (дата Звернення 09.01.2019).
31. Результати дослідження Сайти Вся CRM в Україні. URL: https://www.bitrix24.ua (дата Звернення 10.01.2019).
32. Сьомкіна Т.В., Литвинова О.В., Лобань О.О. Особливості моделей Функціонування IT-компаний в Україні. Науковий вісник Ужгородського національного університету. Серія: Міжнародні економічні відносини та світове господарство. 2018. Вип. 19 (3). С. 84-87.
33. Рейтинг та Каталог CRM-систем популярних в Україні. URL: https: //срм.укр (дата Звернення 10.01.2019).
34. Бутенко Н.В. Впровадження Концепції CRM на промисловому Сайти Вся. Економіка та держава. 2011. № 3. С. 40-42.
35. Ушакова І.О. Соціальні мережі, як засіб впліву на Взаємовідносини з клієнтами. Системи ОБРОБКИ информации. 2012. Вип. 8. С. 54-58.

# ДОДАТКИ

Додаток А

#views.py

**from** django.shortcuts **import** render

**from** django.http **import** HttpResponse

*# Create your views here.*

**from** .models **import** \*

**def** home(request):

 orders = Order.objects.all()

 customers = Customer.objects.all()

 total\_customers = customers.count()

 total\_orders = orders.count()

 refused\_orders=orders.filter(status='refused').count()

 pending\_orders=orders.filter(status='pending').count()

 active\_orders=orders.filter(status='started').count()

 context = {'orders':orders, 'customers':customers,

 'total\_orders':total\_orders, 'refused\_orders':refused\_orders,

 'pending\_orders':pending\_orders, 'active\_orders':active\_orders,

 'total\_customers':total\_customers}

 **return** render(request, 'accounts/dashboard.html', context)

**def** products(request):

 products = Product.objects.all()

 **return** render(request, 'accounts/products.html', {'products':products})

**def** customer(request, pk):

 customer = Customer.objects.get(id=pk)

 orders = customer.order\_set.all()

 orders = orders.objects.order\_by('amount')

 order\_count = orders.count()

 context = {'customer':customer, 'orders':orders,'order\_count':order\_count}

 **return** render(request, 'accounts/customer.html', context)

Додаток Б

#models.py

**from** django.db **import** models

*# Create your models here.*

**class** Customer(models.Model):

 firstname = models.CharField(max\_length=200, null=True)

 secondname = models.CharField(max\_length=200, null=True)

 surname = models.CharField(max\_length=200, null=True)

 phone = models.CharField(max\_length=200, null=True)

 email = models.CharField(max\_length=200, null=True)

 date\_created = models.DateTimeField(auto\_now\_add=True)

 **def** \_\_str\_\_(self):

 **return** str(self.id)+'. '+self.surname+' '+self.firstname+' '+self.secondname

**class** Product(models.Model):

 CATEGORY = (

 ('bail', 'bail'),

 ('non-bail', 'non-bail'),

 )

 name = models.CharField(max\_length=200, null=True)

 percent = models.FloatField(null=True)

 category = models.CharField(max\_length=200, null=True, choices = CATEGORY)

 descriptions = models.CharField(max\_length=200, null=True)

 date\_created = models.DateTimeField(auto\_now\_add=True)

 **def** \_\_str\_\_(self):

 **return** str(self.id)+'. '+self.name+' '+self.descriptions

**class** Order(models.Model):

 STATUS = (

 ('pending', 'pending'),

 ('refused', 'refused'),

 ('started', 'started'),

 ('finished', 'finished'),

 )

 amount = models.IntegerField(null=True)

 term = models.IntegerField(null=True)

 customer = models.ForeignKey(Customer,null = True, on\_delete = models.SET\_NULL)

 product = models.ForeignKey(Product,null = True, on\_delete = models.SET\_NULL)

 date\_created = models.DateTimeField(auto\_now\_add=True)

 status = models.CharField(max\_length=200, null=True, choices = STATUS)