Форма № 24

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій
(повне найменування інституту, факультету)

Кафедра програмних засобів
(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)
бакалавр
(ступінь вищої освіти)

на тему ПРОГРАМНА РЕАЛІЗАЦІЯ ЗАСТОСУНКУ ДЛЯ АВТОМАТИЗАЦІЇ ЗАВДАНЬ РЕКРУТИНГУ.
SOFTWARE IMPLEMENTATION OF THE RECRUITMENT AUTOMATION APPLICATION

Виконала: студент(ка) 4 курсу, групи КНТ-138
Спеціальності 121 Інженерія програмного забезпечення
(код і найменування спеціальності)

Освітня програма (спеціалізація)
Інженерія програмного забезпечення

Пархоменко В.В.
(прізвище та ініціали)
Керівник Олійник А.О.
(прізвище та ініціали)
Рецензент Гофман Є.О.
(прізвище та ініціали)

2022

Форма № 25

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
(повне найменування закладу вищої освіти)

Інститут, факультет ФКНТ
Кафедра програмних засобів
Ступінь вищої освіти бакалавр
Спеціальність 121 Інженерія програмного забезпечення
(код і найменування)
Освітня програма (спеціалізація) Інженерія програмного забезпечення
(назва освітньої програми (спеціалізації)

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
_____С.О. Субботін
“____” _____2022 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

Пархоменко Вікторії Валеріївни
(прізвище, ім’я, по батькові)

1. Тема проєкту (роботи) Програмна реалізація застосунку для автоматизації завдань рекрутингу. Software Implementation of the Recruitment Automation Application

керівник проєкту (роботи) Олійник Андрій Олександрович, д.т.н., професор ,
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)
затверджені наказом закладу вищої освіти від “ 27 ” квітня 2022 року № 102
2. Строк подання студентом проєкту (роботи) 07 червня 2022 року
3. Вихідні дані до проєкту (роботи) технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Проєктування та структура застосунку. 3. Розробка програмного продукту. 4. Експлуатація та тестування програмного продукту.

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)
Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	Олійник А.О., професор		
Нормоконтроль	Липовець М.В., асистент		

7. Дата видачі завдання «04» квітня 2022 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи.	1 тиждень	Завдання, технічне завдання
2	Аналіз предметної області.	1 тиждень	Розділ 1
3	Вибір мови програмування та інших технологій	2 тиждень	Розділ 2
	розробки.		
4	Проєктування та структура застосунку.	2 тиждень	Розділ 2
5	Розробка програмного продукту.	3-4 тижні	Розділ 3
6	Експлуатація та тестування програмного продукту.	5 тиждень	Розділ 4
7	Оформлення пояснювальної записки та документів	6 тиждень	Додатки
	до неї.		
8	Нормоконтроль та рецензування.	7 тиждень	
9	Захист роботи.	8 тиждень	

Студент(ка) _____ Пархоменко В.В.
(підпис) (прізвище та ініціали)

Керівник проєкту (роботи) _____ Олійник А.О.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра: 99 с., 39 рис., 13 табл., 3 дод., 35 джерел.

АВТОМАТИЗАЦІЯ ЗАВДАНЬ РЕКРУТИНГУ, ВЕБЗАСТОСУНОК, ВАКАНСІЯ, КАНДИДАТ, ЗАЯВКА НА ВАКАНСІЮ, ЗАМОВНИКИ, ASP.NET CORE, MICROSOFT SQL SERVER, ENTITY FRAMEWORK CORE, MVC.

Об’єкт дослідження – процес автоматизації завдань рекрутингу.

Предмет дослідження – вебзастосунок, що використовується для автоматизації завдань рекрутингу.

Мета роботи – програмна реалізація вебзастосунку для автоматизації завдань рекрутингу, що дозволить фахівцям у сфері підбору персоналу підвищити ефективність роботи та скоротити час на виконання щоденних операцій.

Матеріали, методи та технічні засоби: платформа розробки вебзастосунків ASP.NET Core, мова програмування C#, реляційна СКБД Microsoft SQL Server, платформа Entity Framework Core, IDE Microsoft Visual Studio 2022 та ноутбук з процесором Intel i5 під керуванням операційної системи Microsoft Windows 10.

Результати. Створено вебзастосунок для автоматизації завдань рекрутингу. Вебзастосунок має клієнт-серверну архітектуру та побудований за архітектурним шаблоном MVC.

Висновки. Розроблений вебзастосунок для автоматизації завдань рекрутингу дозволяє фахівцям у сфері підбору персоналу мінімізувати зайві часові витрати та підвищити продуктивність роботи шляхом автоматизації завдань.

Галузь використання – компанії або підприємства будь-якої спеціалізації.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 99 pages, 39 figures, 13 tables, 3 appendixes, 35 sources.

AUTOMATION OF RECRUITMENT TASKS, WEB APPLICATION, VACANCY, CANDIDATE, REQUEST ON THE VACANCY, CUSTOMERS, ASP.NET CORE, MICROSOFT SQL SERVER, ENTITY FRAMEWORK CORE, MVC.

The object of research is the process of automating recruitment tasks.

The subject of the research is a web application used to automate recruitment tasks.

The purpose of the work is the software implementation of a web application to automate recruitment tasks, which will allow specialists in the field of recruitment to increase efficiency and reduce the time to perform daily operations.

Materials, methods and tools: ASP.NET Core web development platform, C# programming language, relational DBMS Microsoft SQL Server, Entity Framework Core platform, IDE Microsoft Visual Studio 2022 and a laptop with an Intel i5 processor running operating system Microsoft Windows 10.

Results. A web application has been created to automate recruitment tasks. The web application has a client-server architecture and is built on the MVC architectural template.

Conclusions. The developed web application for automation of recruitment tasks allows specialists in the field of personnel selection to minimize unnecessary time and increase productivity by automating tasks.

The area of use is companies or enterprises of any specialization.

ЗМІСТ

С.

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

ATS – Applicant Tracking System;

CSS – Cascading Style Sheets;

DBMS – Database Management System;

HTML – HyperText Markup Language;

IDE – Integrated Development Environment;

JS – JavaScript;

MVC – Model-View-Controller;

SQL – Structured Query Language;

РМ – реляційна модель;

СКБД – система керування базами даних.

ВСТУП

Персонал будь-якої компанії є тією рушійною силою, що дозволяє компанії розвиватися, бути ефективною, сучасною, привабливою для стейкхолдерів, створювати потрібні на ринку продукти чи надавати актуальні послуги, рухатися вперед врешті решт. Саме тому менеджери все більше уваги починають приділяти процесам з управління персоналом. Серед усіх процесів з управління персоналом, найбільш важливим і відповідальним є процес рекрутингу персоналу [1].

Останнім часом всі складові рекрутингу персоналу видозмінюються, що породжено, переважно, динамічним розвитком диджитал-технологій. Диджитал-інновації дозволяють зовсім по-іншому поглянути на процес залучення компетентних спеціалістів і значно підвищують ефективність всього процесу рекрутингу персоналу [1].

Сьогодні використання застарілого обладнання та програмного забезпечення – це величезна перешкода для найму найбільш талановитих і успішних кандидатів, адже вони вимагають дещо іншого підходу до найму. Якщо рекрутери не зможуть адаптувати свої методи пошуку і відбору кандидатів під умови сучасного світу, то дійсно хороших співробітників їм знайти буде складно. Більш того, утримувати та залучати найкращих співробітників без використання диджитал-інновацій також просто неможливо [2].

Актуальність дипломної роботи обумовлена у тому, що сьогодні великий обсяг інформації перевищує можливості рекрутерів для обробки та значно збільшує робоче навантаження. Рішенням даної проблеми є впровадження програмних продуктів для автоматизації завдань з підбору персоналу.

Об’єктом дослідження є процес автоматизації завдань рекрутингу.

Предметом дослідження є вебзастосунок, що використовується для автоматизації завдань рекрутингу.

Метою роботи є програмна реалізація вебзастосунку для автоматизації завдань рекрутингу, що дозволить фахівцям у сфері підбору персоналу підвищити ефективність роботи та скоротити час на виконання щоденних операцій.

Для досягнення мети роботи у дипломній бакалаврській роботі необхідно розв’язати наступні завдання:

– проаналізувати аналогічні програмні продукти;

– розробити технічне завдання;

– обрати мову програмування, середовище для розробки та СКБД;

– проаналізувати та зробити проєктування функціональних вимог до вебзастосунку для автоматизації завдань рекрутингу;

– зробити проєктування архітектури та бази даних вебзастосунку для автоматизації завдань рекрутингу;

– розробити вебзастосунок для автоматизації завдань рекрутингу;

– протестувати програмний продукт.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Опис розглядаємої предметної області

У нинішніх умовах існування бізнесу головною цінністю є кваліфіковані кадри. Неякісні процеси добору та залучення персоналу, як наслідок, приносить значні збитки будь-якому підприємству, тому означене питання є одним із ключових у будь-якій компанії. Останнім часом поряд із терміном «професійний добір» науковці та практики все частіше почали вживати термін «рекрутинг персоналу» [1].

Рекрутинг – це процес пошуку та відбору персоналу згідно встановлених вимог посади та бачення керівництва організації, з урахуванням особистих та професійних якостей кандидатів [3].

Рекрутинг персоналу можна розглядати як бізнес-процес – послідовність виконання періодично повторюваних, цілеспрямованих і регламентованих видів робіт, під час яких вхідні ресурси перетворюються на результат діяльності (процесу) [2].

Процедура рекрутування залежить від різних чинників, насамперед від специфіки організації та вакансії, її місця в організаційній структурі, терміновості заповнення, виділених фінансових ресурсів, наявності на ринку праці достатньої кількості необхідних фахівців [2].

Рекрутинг – це робота з відбору та підбору персоналу, а рекрутер – це людина, що відповідає за пошук потрібних спеціалістів. Рекрутер знаходить кваліфікованих кандидатів на вакансію і працює, щоб задовольнити вимоги як роботодавця, так і працівника протягом усього процесу найму [1].

Фахівці у сфері підбору персоналу зазвичай виконують такі завдання у процесі рекрутингу персоналу:

– організаційне і документаційне забезпечення цього процесу;

– розроблення загальних правил, методів і процедур;

– розроблення компетенцій і формування кваліфікаційних вимог до кандидатів на вакантну посаду;

– визначення джерел залучення та методів оцінювання кандидатів залежно від вакансії, коштів, кон’юнктури ринку праці тощо;

– збір інформації про кандидатів: аналіз резюме, організація анкетування, аналіз документів, перевірка рекомендацій тощо;

– організація та проведення інтерв’ю з кандидатами;

– організація та проведення тестування, використання інших методів оцінювання кандидатів;

– пошук кандидатів на вакантну посаду;

– аналіз ефективності використання різних джерел і процедур добору персоналу тощо [2].

Фундаментальною складовою процесів сучасного рекрутингу персоналу є адаптація до численних сучасних технологій, які дозволяють суттєво підвищити ефективність рекрутингу [1].

Основні чинники впливу на ефективність рекрутингу персоналу із врахуванням диджитал-інновацій:

– мобільні технології;

– технології штучного інтелекту;

– хмарні технології;

– позитивний бренд роботодавця [1].

Серед сучасних цифрових інструментів для рекрутерів можна виокремити такі основні:

– системи управління для автоматизації рекрутингу;

– інструменти управління компенсаціями;

– хмарні інструменти [4].

Для автоматизації завдань рекрутингу застосовуються системи з трекінгу кандидатів або системи для автоматизації рекрутингу [5].

ATS (Applicant Tracking System) – система з управління кандидатами, що дозволяє автоматизувати рішення рекрутингових завдань. ATS може являти собою як класичний додаток, так і онлайн-сервіс (в залежності від потреб та розміру компанії) [5].

До основних завдань систем з управління кандидатами можна віднести наступні:

– обробка вхідного потоку резюме і їх зберігання;

– співвіднесення резюме, що надійшли з вакансією, на яку вони були спрямовані;

– розміщення оголошень про вакансії;

– збір відгуків на вакансії з багатьох джерел;

– індивідуалізована робота з кандидатами в процесі найму;

– робота з керівниками, які наймають;

– робота із зовнішніми провайдерами (рекрутинговими агентствами), які завантажують резюме кандидатів, яких вони представляють, безпосередньо в ATS компанії-клієнта;

– робота з соціальними і професійними мережами;

– робота з пасивними кандидатами, інформація про яких зберігається в ATS;

– збір великого масиву даних про кандидатів [5].

ATS дають компаніям наступні переваги:

– підвищення ефективності обробки вхідних резюме при збереженні високого рівня якості;

– економія часу рекрутерів;

– підвищення привабливості бренду роботодавця;

– можливість використання зібраних даних про кандидатів для підвищення ефективності найму, утримання співробітників, формування подальших стратегій пошуку персоналу [5].

Головне призначення систем – зменшення фінансових та часових затрат на виконанні рутинних завдань. Вони автоматично фільтрують заявки за ключовими словами, навичками, досвіду та освіті кандидата. Якщо рекрутери у своїй роботі використовують ATS, у них автоматично формується кадровий резерв [5].

Використання всіх переваг функцій, доступних в ATS, допоможе скоротити ручні процеси, залучити потенційних кандидатів і, зрештою, швидше заповнити відкриті заявки найкращими кандидатами. Але у багатьох ATS є інші особливості та переваги, які можуть бути не настільки очевидними [5].

Наприклад, система управління кандидатами може бути використана, щоб спонукати людей подавати заявки. Деякі системи використовують чат-ботів для взаємодії з кандидатами та допомоги їм у заповненні заявок. Вона також може намагатися утримувати кандидатів, інформувати претендентів на їх статус і попереджати їх, якщо потрібна додаткова інформація [5].

Система управління кандидатами може використовувати різний рівень інтелекту для сортування кандидатів. Технології варіюються від зіставлення ключових слів до алгоритмів, які глибше аналізують дані кожного кандидата. У відповідь на резюме ATS може надіслати кандидату серію питань, щоб допомогти дізнатися більше про кандидата [5].

1.2 Огляд сучасних рішень вирішення завдання

Для рішення проблеми пов'язаної з автоматизацією завдань рекрутингу з’явилися програмні продукти, які надають можливість прискорити процес пошуку та підбору нових кваліфікованих співробітників.

Нижче будуть розглянуті найпопулярніші системи підбору кандидатів, як «CleverStaff», «Persia» та «Hurma».

1.2.1 Вебзастосунок «CleverStaff»

Однією з популярніших систем підбору пересоналу є «CleverStaff». «CleverStaff» – це автоматизована система управління персоналом як для великого і середнього бізнесу, так і для рекрутерів-фрілансерів. Система

призначена для компаній, що здійснюють підбір співробітників, і для кадрових агентств [6].

Починаючи роботу в системі «CleverStaff», рекрутер отримує доступ до пулу найнеобхідніших інструментів. Розділи-вкладки «Органайзер», «Вакансії», «Кандидати», «Замовники», «Акаунт», «Звіти» – прості та інтуїтивно зрозумілі. Щоб почати роботу, немає необхідності проходити спеціальне навчання [6].

Інтерфейс головної сторінки програми «CleverStaff» зображено на рисунку 1.1.

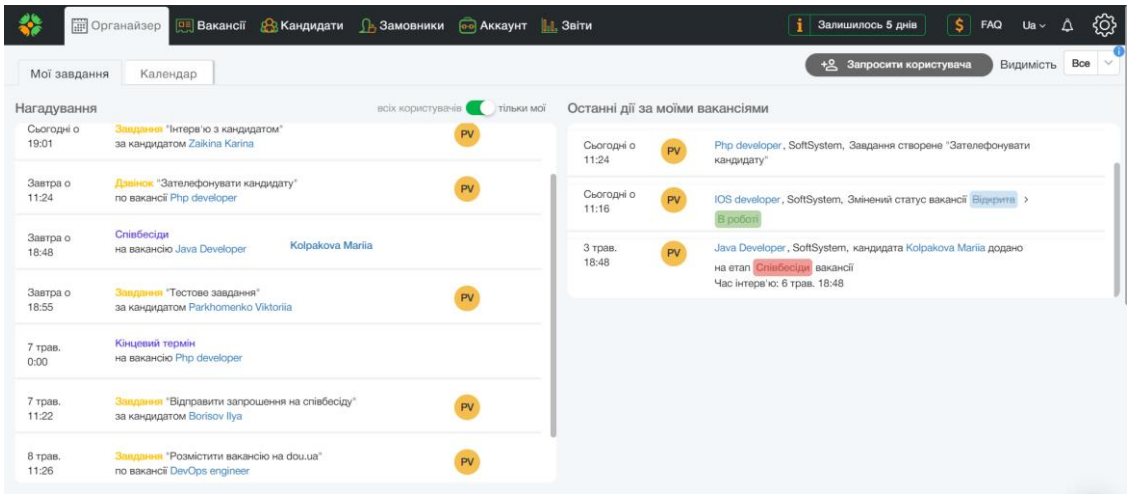

Рисунок 1.1 – Інтерфейс головної сторінки вебзастосунку «CleverStaff»

Основними можливостями, якими володіє система є:

– можливість інтегрувати календар з Google або Outlook Calendar;

– можливість додавання у базу даних вакансії та кандидатів;

– можливість заповнити заявку на вакансію;

– можливість додати кандидатів до вакансії;

– можливість масового видалення кандидатів;

– завантажити резюме у форматі doc, docx, pdf та rtf;

– заповнити вручну картку кандидата;

– імпорт кандидатів з Excel;

– розширення Chrome для швидкого додавання кандидатів з Linkedin, work.ua, rabota.ua;

– міняти налаштування свого акаунта (назва, сфера діяльності, опис компанії);

– створювати кастомні поля (для вкладок «Вакансії», «Кандидати», «Замовники»);

– додавати та редагувати теги [6].

У розділі-вкладці «Звіти» можна переглянути воронку рекрутингу, звіти за джерелами кандидатів, активність користувачів, статистику користувачів, звіт Pipeline.

Воронка рекрутингу будується за кількістю доданих кандидатів в вакансії і переміщень їх по етапах далі. При розрахунку воронки рекрутингу передбачається, що всі етапи вакансії йдуть послідовно, тому вона показує загальну кількість кандидатів, які побували на кожному етапі, а не актуальну кількість кандидатів на етапі [6].

Найцікавіще, що показує воронка рекрутингу, – це конверсія між етапами. Конверсія відображає наскільки ефективні етапи у процентному співвідношенні [6].

Відносна конверсія відображає результати у порівнянні з попереднім етапом. Абсолютна конверсія відображає результати кожного етапу у порівнянні з найпершим етапом вакансії [6].

Гарна конверсія економить багато часу, тому що для закриття вакансії потрібно знаходити менше кандидатів і менше з ними працювати. Етапи з низькою конверсією – це зона росту продуктивності роботи [6].

Корисно порівнювати воронки рекрутингу різних рекрутерів і всієї компанії. Стане видно хто більш ефективний і на яких етапах [6].

Звіт Pipeline допомагає визначити проблемні вакансії і завчасно посилити роботу по ним. Він показує:

– активні вакансії з відповідальними;

– останні 2 етапи, за якими була активність;

– в який день і скільки кандидатів пропрацювали;

– кількість найманих кандидатів і тих, які не підійшли;

– вакансії з працевлаштованими кандидатами;

– вакансії по даті кінцевого терміну;

– вакансії за якими більш ніж 5 днів не було дій [6].

Звіти за джерелами відображають статистику за типами джерел і допомагають зрозуміти, яке джерело дає більший відсоток закритих вакансій. Звіти за джерелами поділяються на звіти за кількістю та ефективністю джерел кандидатів [6].

Звіт за кількістю джерел кандидатів показує статистику в розрізі користувачів і часових інтервалів. Звіт ефективності джерел кандидатів показує джерела, на яких необхідно сконцентруватися, щоб наймати більше працівників. Звіт статистики користувачів показує результати роботи кожного користувача облікового запису [6].

Звіт активності користувачів відображає щотижневу активність різних користувачів для обраних вакансій за певний період часу [6].

Інтерфейс розділів-вкладок «Вакансії» та «Кандидати» зображено на рисунках 1.2 та 1.3.

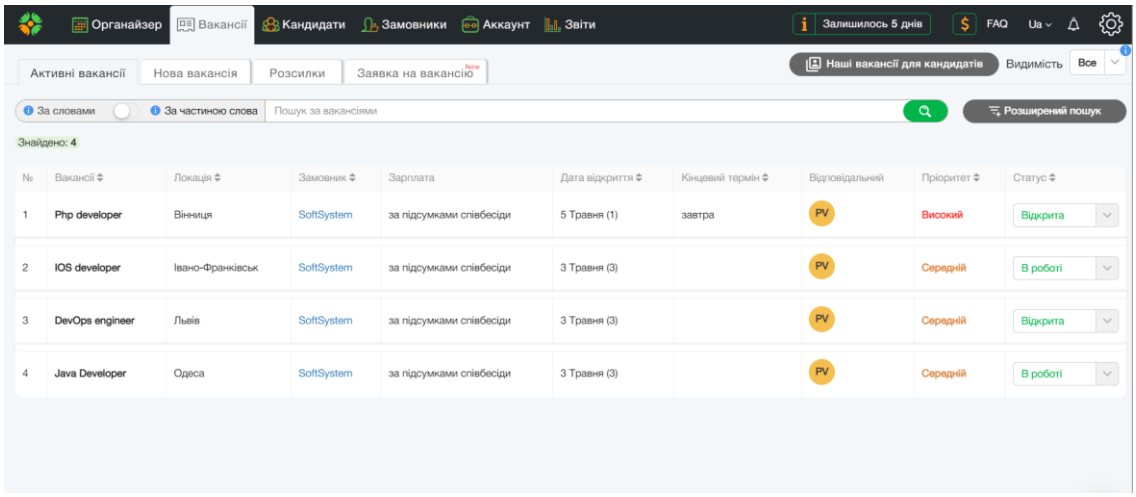

Рисунок 1.2 – Сторінка перегляду активних вакансій вебзастосунку «CleverStaff»

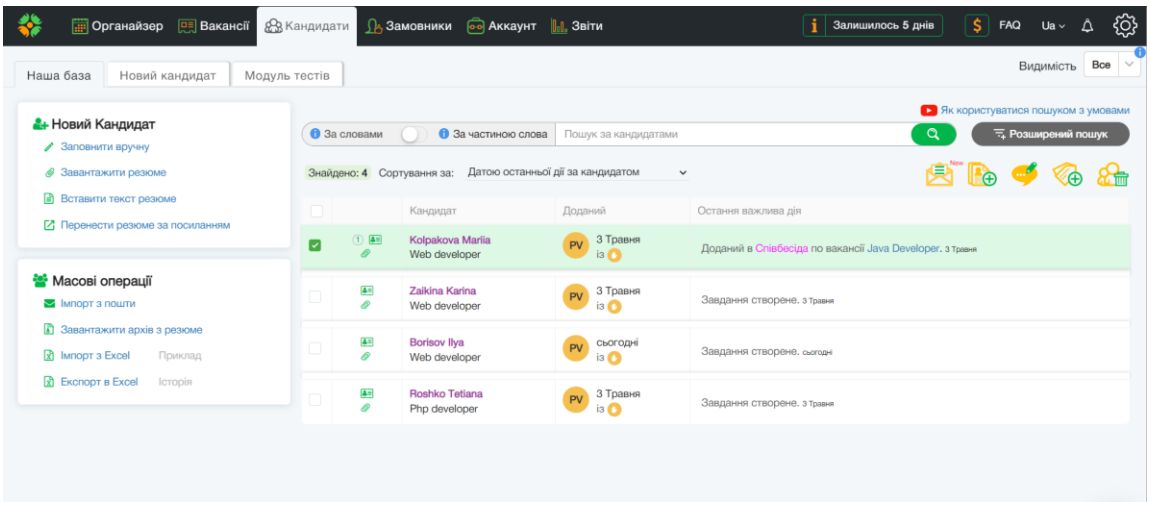

Рисунок 1.3 – Сторінка перегляду бази кандидатів вебзастосунку «CleverStaff»

Серед переваг можна виокремити наступні:

– крім стандартних полів при редагуванні вакансії є можливість додавати свої особисті кастомні поля;

– створювати свій особистий сайт із вакансіями;

– створювати свій набір параметрів, за якими будуть оцінюватися кандидати;

– гнучке управління базою кандидатів;

– має конструктор звітів;

– синхронізація з Google-календарем;

– інтеграція з work.ua, rabota.ua та LinkedIn [6].

Основним недоліком системи є платність.

1.2.2 Вебзастосунок «Persia»

Наступним аналогічним рішенням є «Persia». «Persia» – це онлайн-система для автоматизації роботи команди рекрутерів. Цей програмний продукт допомагає організувати роботу команди рекрутерів і зробити її більш ефективною [7].

Основні функціональні можливості системи:

– база кандидатів та вакансій;

– історія листування;

– пошук за ключовими словами;

– синхронізація з Google календарем;

– публічна сторінка компанії з вакансіями;

– завантаження резюме з сайтів робіт і соцмереж;

– спільний доступ з іншими рекрутерами;

– історія роботи з кандидатами;

– статистика за закритими вакансіями, кількості кандидатів, аналітика якості роботи з порталами пошуку роботи [8].

Серед переваг можно виділити лише одне, що система має розширення для браузерів. Також ресурс постійно оновлюється і додає нові корисні функції. Система безкоштовна тільки для одного користувача і з обмеженими опціями [9].

Недоліки системи:

– відсутній конструктор звітів;

– відсутня Kanban-дошка;

– відсутній мобільний додаток [8].

Інтерфейс сторінок перегляду бази вакансій та кандидатів вебзастосунку «Persia» наведено на рисунках 1.4 та 1.5.

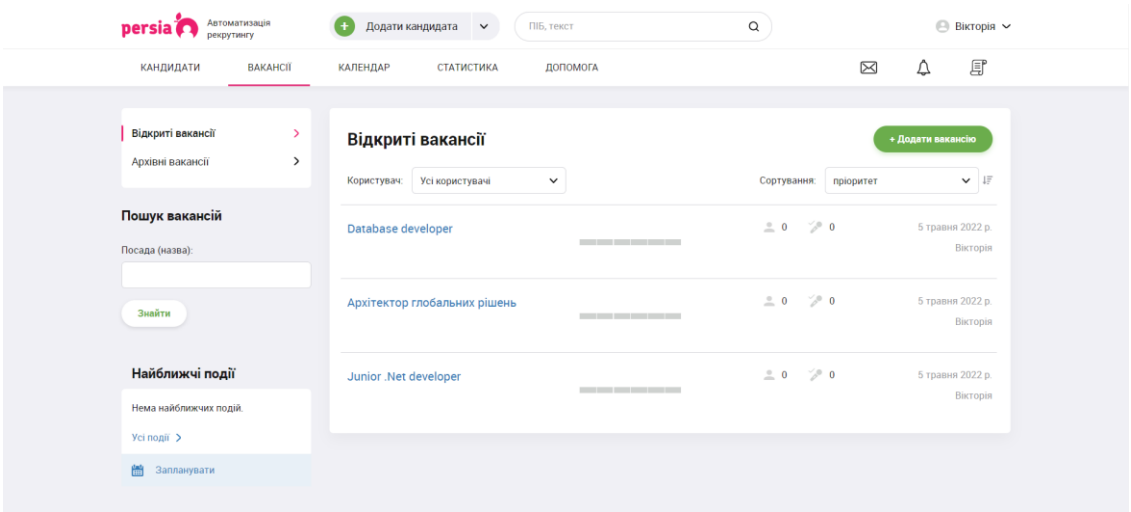

Рисунок 1.4 – Сторінка бази вакансій вебзастосунку «Persia»

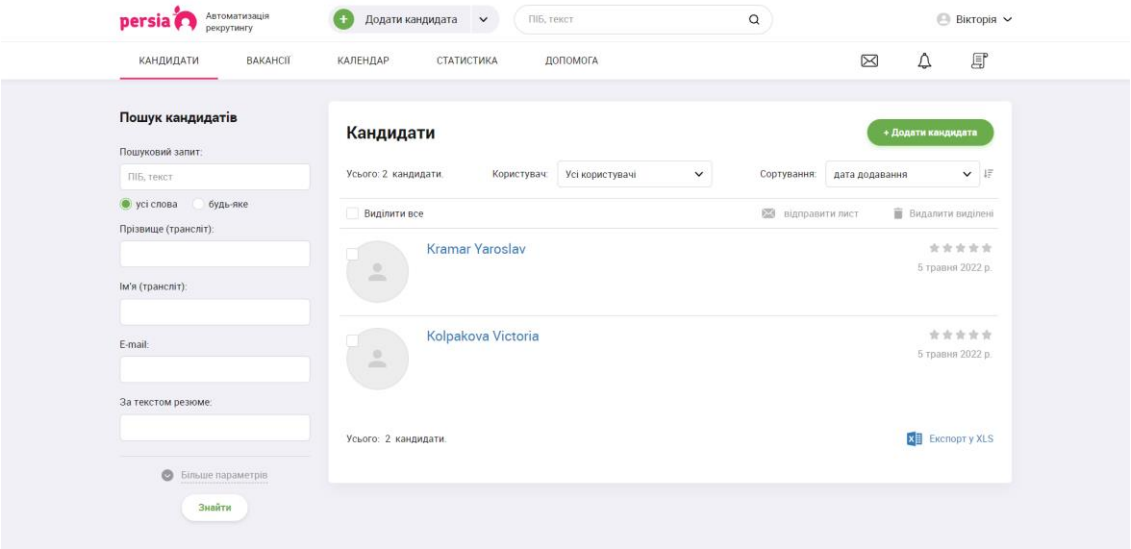

Рисунок 1.5 – Сторінка бази кандидатів вебзастосунку «Persia»

1.2.3 Вебзастосунок «Hurma»

Ще одним існуючим рішенням є вебзастосунок «Hurma». «Hurma» – це перша українська інформаційна система управління персоналом або система управління людськими ресурсами, яка включає в себе автоматизацію процесів рекрутингу та вбудовану систему для відстеження цілей компанії та ключових результатів [9].

Нижче наведені основні можливості системи:

– інтеграція з порталами пошуку роботи;

– воронка рекрутингу;

– база кандидатів та вакансій для рекрутера;

– парсинг резюме з різних типів файлів;

– імпорт кандидатів з інших рекрутингових систем;

– контрольна панель;

– теги для фільтрації кандидатів;

– база співробітників;

– створювати опитування;

– оформити заявку на вакансію;

– аналіз кількості відкритих вакансій за періодами;

– розрахунок коефіцієнту прийняття оферу;

– облік часу [10].

У системі присутня детальна аналітика результатів роботи над вакансією, де воронка рекрутингу показує рух кандидатів по етапах, а також візуально відображає всі причини відмов кандидатів або компанії, що дозволяє проаналізувати цифри і виявити слабкі місця [9].

Штучний інтелект виконує у системі попередження ризику звільнення співробітників та вибір найкращого кандидата на вакансію [9].

На рисунку 1.6 наведена головна сторінка вебзастосунку «Hurma».

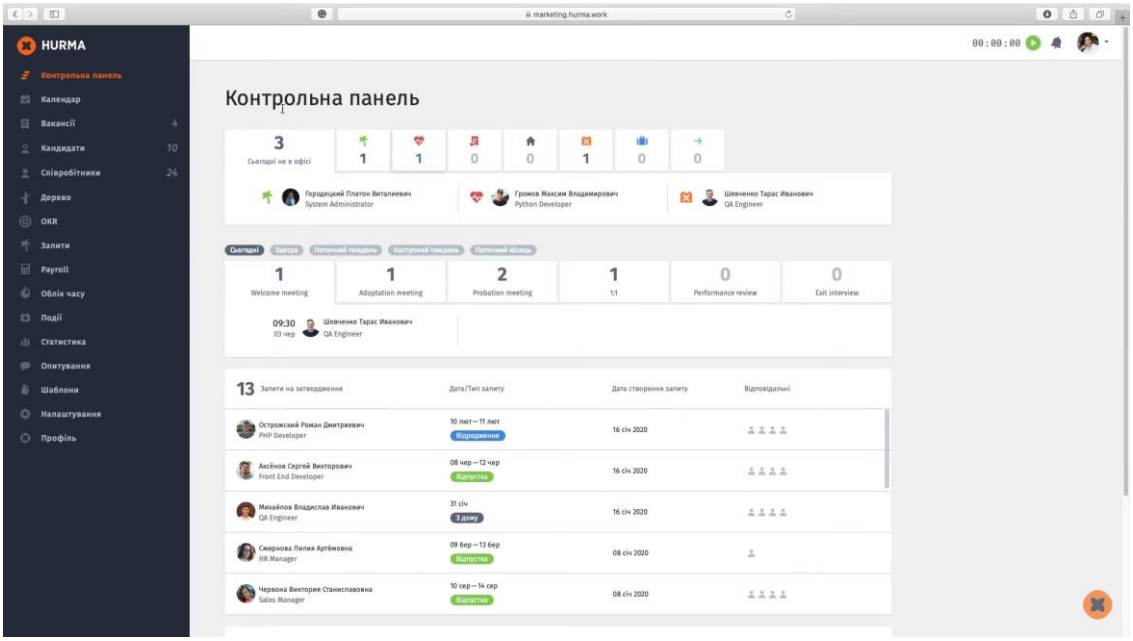

Рисунок 1.6 – Головна сторінка вебзастосунку «Hurma»

Однією з переваг системи над вище розглянутими є Kanban-дошка з поділом кандидатів по кожному з етапів рекрутингу [11].

1.2.4 Аналіз і порівняльна таблиця існуючих рішень

У пунктах вище були наведені найбільш використовувані на сьогоднішній день вебзастосунки, які дозволяють автоматизувати та

прискорити роботу рекрутерів. Розглянуті вебзастосунки мають багато переваг, але у всіх їх присутній головний недолік – платність.

Серед переваг, які присутні для всіх розглянутих вебзастосунків є можливість інтеграції з поштою, календарем, сайтами пошуку роботи та статистика користувачів, яка показує результативність роботи рекрутерів. Також усі вебзастосунки мають можливість створювати особистий сайт із вакансіями, при цьому не залучаючи фахівців з розробки сайтів.

Порівняння сучасних рішень вирішення завдання, які були розглянуті вище наведено в таблиці 1.1.

Таблиця 1.1 – Порівняння сучасних рішень вирішення завдання

Критерій порівняння	«CleverStaff»	«Persia»	«Hurma»
Інтеграція з поштою	+	+	+
Інтеграція з календарем	+	+	+
Інтеграція з сайтами пошуку роботи	+	+	+
Розширення для браузерів	+	+	–
Статистика користувачів	+	+	+
Воронка рекрутингу	+	–	+
Конструктор звітів	+	–	–
Багатомовність	+	+	+
Кастомні поля	+	–	–
Публічна сторінка компанії	+	+	+
Платність	+	+	+

1.3 Постановка завдання роботи

Завданням роботи є програмна реалізація вебзастосунку для автоматизації завдань рекрутингу, тому можна виокремити наступні основні задачі роботи:

– аналіз аналогічних програмних продуктів;

– розробка технічного завдання;

– вибір мови програмування, середовища для розробки та СКБД;

– аналіз вимог до вебзастосунку для автоматизації завдань рекрутингу;

– проєктування вебзастосунку для автоматизації завдань рекрутингу;

– програмна реалізація вебзастосунку для автоматизації завдань рекрутингу;

– тестування програмного продукту.

1.4 Висновки за розділом 1

У цьому розділі розглянуто предметну область та проаналізовано аналогічні програмні продукти для автоматизації завдань рекрутингу. Було наведено основні задачі роботи, які необхідно вирішити.

Розглянувши предметну область можна зробити висновок, що основною метою будь-якого застосунку для автоматизації завдань рекрутингу є пришвидшення роботи рекрутерів та зручність внесення інформації у базу даних. Також такі системи допомагають компаніям швидко аналізувати дані кандидатів, щоб допомогти їм швидко приймати кращі рішення.

Виходячи з виконаного огляду існуючих програмних продуктів, виявлені основні можливості, якими повинна володіти програма, що забезпечує автоматизацію завдань рекрутингу:

– додавати, редагувати та видаляти кандидатів;

– додавати, редагувати та видаляти вакансії;

– створювати та заповнювати заявку на вакансію;

– додавати користувачів;

– додавати замовника за вакансіями.

2 ПРОЄКТУВАННЯ ТА СТРУКТУРА ЗАСТОСУНКУ

2.1 Вибір основних засобів розробки

Мовою для програмування серверної частини було обрано C#. Для реалізації серверної частини було обрано фреймворк розробки вебзастосунків ASP.NET Core та платформу .NET Core версії 6.0.

За допомогою REST API (Representational State Transfer Application Programming Interface) комунікують клієнт з сервером, бо цей спосіб використовує http методи. REST API – спосіб створення або побудови програмного інтерфейсу застосунку за допомогою http протоколу. Цю технологію використовують повсюди, де користувачу сайту або якогось вебзастосунку потрібно отримати дані з серверу [12].

В якості СКБД було обрано Microsoft SQL Server. Також інструмент для роботи з даними Entity Framework Core, щоб прискорити роботу з даними.

2.1.1 Мова програмування C#

C# – сучасна об'єктно-орієнтована мова програмування загального призначення, розроблена компанією Microsoft і схвалена Європейською асоціацією виробників комп'ютерів (ECMA) та Міжнародною організацією зі стандартів (ISO) [13].

C# розроблено для загальномовної інфраструктури (CLI), яка складається з виконуваного коду та середовища виконання, що дозволяє використовувати різні мови високого рівня на різних комп’ютерних платформах та архітектурах [13].

У C# немає глобальних змінних, функцій чи процедур. Це зроблено з метою запобігання конфліктів імен. Програма складається з одного чи більше описів типів – класів, інтерфейсів, структур, перелічень або делегатів. Типи об'єднуються в так звані простори імен, які їх логічно групують [14].

Мова має строгу статичну типізацію, підтримує поліморфізм, перевантаження операторів, вказівники на функції-члени класів, атрибути, події, властивості, винятки, коментарі у форматі XML. Перейнявши багато від своїх попередників – мов C++, Delphi, Модула і Smalltalk – C#, спираючись на практику їхнього використання, виключає деякі моделі, що зарекомендували себе як проблематичні при розробці програмних систем: так, C# не підтримує множинне спадкування класів (на відміну від C++) або виведення типів (на відміну від Haskell) [12].

Зараз на ньому пишуться різні програми: від невеликих десктопних програм до великих вебпорталів і вебсервісів, що обслуговують щодня мільйони користувачів. Об'єктно-орієнтований підхід дозволяє вирішити завдання з побудови великих, але в той же час гнучких, масштабованих та розширюваних додатків. C# продовжує активно розвиватися і з кожною новою версією з'являється все більше цікавих функціональностей [15].

C# практично завжди застосовується в зв’язці з будь-яким фреймворком, що і впливає на поведінку мови. Саме це і дозволяє використовувати C# в самих різних сферах [16].

До переваг мови над іншими можна віднести наступні:

– мова C# строго типізована, значить її простіше опанувати початківцям;

– мова C# універсальна та багатопарадигмальна;

– охоплює дисципліни програмування зі статичною типізацією, сильною типізацією, лексичною, імперативною, декларативною, функціональною, загальною, об’єктно-орієнтованою (на основі класів) і компонентно-орієнтованою;

– мову C# також можна використовувати для широкого спектру додатків, включаючи корпоративне програмне забезпечення, мобільні додатки тощо;

– мова програмування постійно розвивається та додаються корисні функції;

– C# у порівнянні з Java легше взаємодіє, з кодом програм, написаних на інших мовах;

– синтаксис досить мінімалістичний;

– ручне управління пам’яттю [17].

Порівняння мови програмування C# з іншими мовами програмування наведено в таблиці 2.1.

Таблиця 2.1 – Порівняння мов програмування

Критерій порівняння	C#	PHP	Python	Ruby
Багатогранність розробки	++	–	+	+
Статистична типізація	+	–	–	–
Легкість освоєння	+	–	+	+
Багатопарадигмальна мова	++	+	+	+
Компільований тип мови	+	–	–	–
Мова високого рівня	+	+	+	+
Об’єктно-орієнтована мова	+	+	+	+
Швидкість виконання	+	+	–	–
Мультипотоковість	+	–	+	+
Модифікатор доступу до методу	+	+	–	+
Асинхронність	+	–	+	–

Ще однією перевагою мови програмування C# є те, що програми, скомпільовані в рідний код під час компіляції, як правило, працюють швидше, ніж ті, що перекладаються під час виконання [16].

2.1.2 Фреймворк ASP.NET Core

ASP.NET Core – це фреймворк від компанії Microsoft, який використовує середовище виконання .NET Core. Він призначений для розробки якісних сучасних вебзастосунків та є продовженням розвитку платформи ASP.NET. Однак це не просто оновлена технологія. Своїм виходом ASP.NET Core фактично позначив якісну зміну усієї платформи. На сьогоднішній день цей фреймворк є дуже затребуваним у розробці. По суті,

це повна трансформація ASP.NET, яка об'єднує MVC структуру і вебслужбу в єдину платформу [18].

Фреймворк ASP.NET Core дозволяє розробникам з легкістю налаштовувати параметри безпеки застосунків і керувати ними. ASP.NET Core включає функції для управління аутентифікацією, авторизацією, захистом даних, роботи інтернет-протокол, захистом від підробки запитів і роботи з спільним використанням ресурсів між різними джерелами. Такі функції безпеки дозволяють створювати надійні і безпечні ASP.NET Core застосунки [18].

Завдяки MVC розробка вебзастосунків стає простіше, а робочий процес – більш ефективним. ASP.NET Core значно спрощує розробку, компіляцію і тестування розробниками в моделі, контролері або показі [18].

ASP.NET Core дозволяє створювати вебзастосунки за допомогою різних моделей розробки. ASP.NET Core MVC представляє в загальному вигляді побудову додатків навколо трьох основних компонентів – Model (моделі), View (представлення) і Controller (контроллери), де моделі відображають роботу з даними, контролери представляють логіку обробки запитів, представлення визначення візуального складу [19].

Pages – це основний елемент ASP.NET Core, який робить сценарії програмування на основі вебсторінок більш продуктивними. За допомогою Razor Pages кожна вебсторінка стає автономною завдяки View компоненту, тоді як код зберігає стабільну структуру [19].

Також ASP.NET підтримує інтеграцію з сучасними фреймворками інтерфейсу користувача, такі як AngularJS, ReactJS, Umber, Bootstrap тощо, і керувати ними, використовуючи Bower. JavaScriptServices допомагають створювати багатофункціональні інтерфейсні вебзастосунки [19].

Застосунки на базі ASP.NET Core можуть працювати в операційних системах Windows, Linux і Mac [19].

Вебзастосунок ASP.NET Core може розміщуватися на кількох платформах із будь-яким вебсервером, таким як IIS (Internet Information Services), Apache тощо [19].

Порівняння серверних фреймворків ASP.NET Core та Laravel наведено в таблиці 2.2.

Таблиця 2.2 – Порівняння серверних фреймворків

Критерій порівняння	ASP.NET Core	Laravel
Мультиплатформеність	+	+
Будь-яка мова програмування	+	–
Велика кількість документації	+	+
Високий рівень безпеки	+	–
Придатність до великих розмірів рішень	+	–
Архітектурний шаблон MVC	+	+
Висока продуктивність роботи	+	–
Компільований тип коду	+	–
Модульний підхід розробки	+	+

ASP.NET Core надає такі переваги:

– Razor Pages робить кодування сценаріїв, орієнтованих на сторінку, простішим і продуктивнішим;

– Blazor дозволяє використовувати C# у браузері разом із JavaScript;

– можливість розробки та роботи на Windows, macOS та Linux;

– інтеграція сучасних фреймворків на стороні клієнта та робочих процесів розробки;

– хмарна система конфігурації на основі середовища;

– вбудована ін'єкція залежностей [19].

Деякі з значних переваг фреймворку ASP.NET Core перед платформою ASP.NET:

– висока продуктивність;

– міжплатформенність;

– відкритий вихідний код [19].

2.1.3 Система керування базами даних

База даних – сукупність даних, організованих відповідно до концепції, яка описує характеристику цих даних і взаємозв'язки між їх елементами [20].

Сутність – це будь-який об'єкт в базі даних, який можна виділити виходячи з суті предметної області, для якої розробляється ця база даних [20].

Система керування базами даних (СКБД) – це програма, яка взаємодіє з кінцевими користувачами, іншими програмами та самою базою даних для збору й аналізу даних. СКБД загального призначення дозволяє визначати, створювати, робити запити, оновлювати та адмініструвати бази даних [21].

Зазвичай СКБД складається з:

– даних;

– програмне забезпечення;

– апаратне забезпечення;

– процедури;

– мова доступу до бази даних;

– ядро баз даних;

– словник даних;

– процесор запитів;

– менеджери (дані, бази даних та інші) [21].

Дані є найважливішим компонентом СКБД. Це вихідний матеріал, з якого генерується інформація. Основним завданням СКБД є опрацювання даних. База даних містить як метадані, так і фактичні (або експлуатаційні) дані. Метадані – це опис даних або даних про дані [21].

Програмне забезпечення – це набір програм, які використовуються для контролю та керування загальною базою даних. Програмне забезпечення СКБД забезпечує простий у використанні інтерфейс для зберігання,

отримання та оновлення даних у базі даних. Цей компонент використовується для обробки мови доступу до бази даних і перетворює його в фактичні команди бази даних для виконання або запуску в базу даних [21].

Компонент СКБД апаратне забезпечення складається з набору фізичних електронних пристроїв, таких як комп'ютери (персональні комп'ютери, робочі станції, сервери та суперкомп'ютери), канали вводу або виводу для даних, пристрої зберігання даних, мережеві пристрої (концентратори, комутатори, маршрутизатори, волоконно-оптичні) та будь-які інші залучені фізичні компоненти до того, як будь-які дані будуть успішно збережені в пам'яті. Це всі ті фізичні компоненти, які створюють інтерфейс між комп'ютерами та користувачами. Цей компонент СКБД використовується для збереження та зберігання даних у базі даних [21].

Процедури відносяться до загальних правил та інструкцій, які допомагають розробити базу даних і використовувати СКБД. Використовується для створення резервної копії бази даних та зміни структури бази даних тощо. Процедури також використовуються для забезпечення, що існує організований спосіб моніторингу та аудиту як даних, які надходять до бази даних, так і інформації, яка створюється за допомогою цих даних [21].

Мова доступу до бази даних – це проста мова, призначена для написання команд для доступу, вставки, оновлення та видалення даних, що зберігаються в будь-якій базі даних. Кожен користувач пише набір відповідних команд мовою доступу до бази даних, подає їх у СКБД, яка потім обробляє дані та генерує їх, виводить набір результатів у доступну для читання форму. Користувач може створювати нові бази даних, таблиці, вставляти дані, отримувати збережені дані, оновлювати дані та видаляти дані за допомогою мови доступу [21].

Більшість систем управління базами даних містять власний інтерфейс прикладного програмування, який дозволяє користувачеві взаємодіяти з базовим механізмом, не переходячи через інтерфейс користувача СКБД.

Механізми баз даних є досить складними та використовують спеціальні структури індексації для прискорення пошуку даних [21].

Система керування реляційною базою даних є механізмом баз даних на основі РМ. РМ для СКБД – це підхід до роботи з даними на основі алгебри відношень (кортежі, логіка предикатів, структури, відносини) [21].

Коротко особливості реляційних баз даних можна сформулювати так:

– дані зберігаються в таблицях, що складаються зі стовпців і рядків;

– на перетині кожного стовпця та рядка є рівно одне значення;

– кожен стовпець має ім'я, яке служить його ім'ям, і всі значення в одному стовпці є одного типу;

– запити до бази даних повертають результати у вигляді таблиць, які також можуть діяти як об’єкт запиту;

– рядки в реляційній базі даних не впорядковані: упорядкування виконується в момент формування відповіді на запит;

– загальноприйнятим мовним стандартом для роботи з реляційними базами даних є мова SQL [21].

До переваг РМ можна виокремити наступне:

– простота та доступність для розуміння користувачами;

– суворі правила проєктування засновані на математичній теорії;

– при зміні реляційної бази даних відповідні зміни в додатках мінімальні;

– для організації запитів і написання прикладного програмного забезпечення не потрібно знати конкретну організацію бази даних у зовнішній пам'яті [21].

До недоліків РМ можна виокремити наступне:

– не завжди предметну область можна представити у вигляді таблиць;

– в результаті логічного проєктування виникає набір таблиць, що призводить до труднощів у розумінні структури даних;

– база даних займає відносно великий обсяг зовнішньої пам'яті;

– відносно низька швидкість доступу до даних [21].

Мова структурованих запитів (SQL) – це стандартна комп’ютерна мова для системи керування реляційними базами даних і маніпулювання даними. SQL призначена для керування даними, які зберігаються в системі керування реляційною базою даних, або для потокової обробки в системі керування реляційними потоками даних [21].

SQL складається з:

– мови запитів даних;

– мови визначення даних;

– мови керування даними;

– мови маніпулювання даними [21].

Однією з найбільших переваг використання СКБД є те, що вона дає можливість кінцевим користувачам і програмістам отримати доступ і використовувати одні й ті ж дані одночасно, керуючи цілісністю даних. Дані краще захищаються та обслуговуються, коли ними можна спільно використовувати СКБД замість створення нових ітерацій тих самих даних, що зберігаються в нових файлах, для кожної нової програми. СКБД забезпечує центральне сховище даних, до якого можуть контролюватися декілька користувачів [22].

Ще одна перевага СКБД полягає в тому, що адміністратори баз даних можуть використовувати її для встановлення логічної структурованої організації даних. СКБД забезпечує економію масштабу для обробки великих обсягів даних, оскільки вона оптимізована для таких операцій [23].

Microsoft SQL Server – це реляційна СКБД. У реляційних базах даних дані зберігаються в таблицях. Взаємопов'язані дані можуть групуватися в таблиці, крім того, можуть бути встановлені також і взаємини між таблицями. Користувачі отримують доступ до даних на сервері через програми, а адміністратори, виконуючи завдання конфігурування, адміністрування та підтримки бази даних, виробляють безпосередній доступ до сервера. Microsoft SQL Server є масштабованої базою даних, це означає,

що вона може зберігати значні обсяги даних і підтримувати роботу багатьох користувачів, які здійснюють одночасний доступ до бази даних [22].

Microsoft SQL Server не призначений безпосередньо для розробки користувальних додатків, а виконує функції керування базою даних. Сервер має засоби віддаленого адміністрування і керування операціями, організовані на базі об'єктно-орієнтованої розподіленого середовища управління [22].

Microsoft SQL Server призначений винятково для підтримки систем, що працюють у середовищі клієнт-сервер. Він підтримує широкий спектр засобів розробки і максимально простий в інтеграції з додатками, що працюють на персональному комп'ютері [23].

Microsoft SQL Server містить асистент адміністратора. Цей інструмент дозволяє призначати основні процедури супроводу бази даних і визначати для них графік виконання. Операції по супроводу баз даних включають перевірку розподілу сторінок, цілісності покажчиків у таблицях (включаючи системні) і індексах, відновлення інформації, необхідної оптимізатору, реорганізацію сторінок у таблицях і індексах, створення страхувальних копій таблиць і журналів транзакцій. Всі ці операції можуть бути встановлені для автоматичного виконання по заданому адміністратором графіку [23].

Одним з головних подій, що визначили подальшу долю Microsoft SQL Server, стало рішення Microsoft зосередити зусилля на підтримці платформи Windows. Ця СКБД настільки пов'язана з операційною системою, що її надійність, масштабованість і продуктивність визначаються надійністю, масштабованістю і продуктивністю самої платформи [23].

Чим ширше використовуються розподілені обчислення, тим більш важливою виявляється можливість зберігати дані де завгодно, зокрема на робочій станції або портативному комп'ютері. Незважаючи на твердження деяких аналітиків про те, що в епоху Інтернет-додатків настільні СКБД вже не потрібні, вони як і раніше широко застосовуються у всіх сферах бізнесу. Microsoft SQL Server можна застосовувати на будь-яких Intel-сумісних комп'ютерах під управлінням Windows будь-якої версії [23].

Однією з переваг Microsoft SQL Server є простота його застосування, зокрема адміністрування [23].

Порівняння СКБД Microsoft SQL Server та PostgreSQL наведено в таблиці 2.3.

Таблиця 2.3 – Порівняння СКБД

Критерій порівняння	Microsoft SQL Server	PostgreSQL
Процедурні розширення SQL	Transact-SQL	PL/pgSQL
Безпека даних	+	–
Легкість освоєння	+	–
Велика кількість документації	+	–
Велика кількість підтримуваних типів даних	+	+
Зручний інтерфейс	+	–
Реляційна модель даних	+	+
Масштабованість	+	–
Клієнт-серверна архітектура	+	+
Необмежений розмір бази даних	+	+
Висока продуктивність роботи	+	–

За даними Transaction Processing Performance Council (TPC), Microsoft SQL Server зараз є рекордсменом по продуктивності [21].

Transact-SQL – це набір розширень програмування від Sybase і Microsoft, які додають кілька функцій до мови структурованих запитів (SQL), включаючи оператори потоку if, while, керування транзакціями, обробку винятків і помилок та обробку рядків [21].

Основна відмінність Transact-SQL від SQL, що Transact-SQL містить процедурне програмування та локальну змінну, тоді як SQL цього не має [21].

Найпопулярнішими операторами Transact-SQL є збережена процедура та користувацькі функції, які є скомпільованим та збереженим кодом Transact-SQL [21].

2.1.4 Фреймворк для роботи з даними

Entity Framework Core – це фреймворк об’єктного-реляційного відображення, який дозволяє розробникам працювати з реляційною базою даних. Це дозволяє розробникам працювати з даними як з об’єктами та властивостями. Використовуючи Entity Framework Core, розробники видають запити за допомогою інтегрованого мовного запиту, потім отримують і маніпулюють даними як строго типізовані об’єкти за допомогою мови програмування C# [24].

Платформа дозволяє розробникам працювати з даними, використовуючи об'єкти предметних класів, не зосереджуючи увагу на таблицях і стовпцях бази даних, де ці дані зберігаються [24].

Entity Framework від Microsoft розвинувся на основі методології, відомої як моделювання сутності-відносин. Моделювання сутності-відносин визначає схему сутностей та їх зв’язки один з одним. Сутності визначають схему об'єкта, але не його поведінку [24].

До особливостей платформи відносяться:

– переклад строго типізованих запитів за допомогою мовного інтегрованого запиту;

– візуальний дизайнер для створення моделей сутностей;

– моделі можуть бути створені з існуючих баз даних, а потім відредаговані вручну, або їх можна створити з нуля, а потім використовувати для створення нових баз даних;

– інтеграція з моделями додатків .NET Framework, включаючи ASP.NET;

– підключення до бази даних на основі ADO.NET і численних постачальників, доступних для підключення до SQL Server, Oracle, MySQL, SQLite, PostgreSQL, DB2 тощо [24].

За допомогою Entity Framework розробники можуть працювати на більш високому рівні абстракції, коли вони мають справу з даними, а також

можуть створювати та підтримувати орієнтовані на дані програми з меншою кількістю коду, ніж у традиційних програмах. Оскільки Entity Framework є компонентом .NET Framework, програми Entity Framework можуть працювати на будь-якому комп'ютері, на якому встановлено .NET Framework, починаючи з версії 3.5 SP1 [25].

Тип сутності – фундаментальний будівельний блок для опису структури даних за допомогою моделі даних Entity. У концептуальній моделі типи сутностей будуються з властивостей і описують структуру концепцій верхнього рівня, таких як клієнти та замовлення в бізнес-додатку [25].

Щоб зберегти дані при зміні моделі, у Entity Framework Core існує функція міграції. Вона дозволяє послідовно застосовувати зміни до бази даних, щоб синхронізувати її з моделлю даних [25].

У міграції існують операції, які дозволяють видаляти, додавати стовпці та таблиці, зовнішні ключі, змінювати налаштування стовпців, додавати, видаляти та змінювати дані, і так далі. При створенні міграції автоматично створюється клас, де виконуються операції, які необхідні застосування міграції Up() та її повернення метод Down() [25].

Під час роботи з інструментами моделі даних Entity, концептуальна модель, модель зберігання та відображення між ними виражаються в схемах на основі XML і визначаються у файлах, які мають відповідні розширення імен:

– мова визначення концептуальної схеми (CSDL) визначає концептуальну модель;

– мова визначення схеми сховища (SSDL) визначає модель зберігання;

– мова специфікації відображення (MSL) визначає відображення між сховищем та концептуальними моделями [26].

Entity Framework використовує ці файли моделі та відображення для створення, читання, оновлення та видалення операцій щодо сутностей і зв'язків у концептуальній моделі до еквівалентних операцій у джерелі даних.

Entity Framework навіть підтримує відображення сутностей у концептуальній моделі до збережених процедур у джерелі даних [26].

До переваг технології Entity Framework Core можна віднести наступне:

– незалежність технології Entity Framework Core та реалізації .NET;

– створення бази даних реалізовується написанням об'єктно-орієнтованого коду, тобто на основі абстракції класу технології Entity;

– Entity Framework Core самостійно створює реляційну базу даних;

– технологія Entity Framework Core реалізована таким чином, що взаємодія з базою даних не залежить від реляційної СКБД;

– механізм реалізації SQL-запитів до бази даних у технології Entity Framework Core реалізований через вбудовану мову інтегрованого запиту;

– підтримка запитів з параметрами та роботи з процедурами;

– реалізація підтримки роботи не тільки з типами даних, які є вбудованими, але і підтримка реалізації з типами даних, які можна запрограмувати [26].

2.1.5 Середовище для розробки

Microsoft Visual Studio – це IDE, створена Microsoft і використовується для різних типів розробки програмного забезпечення, наприклад комп’ютерних програм, вебсайтів, вебпрограм, вебсервісів та мобільних додатків. Він містить інструменти завершення, компілятори та інші функції для полегшення процесу розробки програмного забезпечення [27].

Його інтерфейс користувача використовується для розробки програмного забезпечення для редагування, налагодження та створення коду. Visual Studio містить редактор коду, який підтримує IntelliSense (компонент завершення коду), а також рефакторинг коду. Інтегрований налагоджувач працює і як налагоджувач на рівні джерела, і як налагоджувач на рівні

машини. Інші вбудовані інструменти включають профайлер коду, конструктор для створення додатків із графічним інтерфейсом користувача, вебдизайнер, конструктор класів і дизайнер схем баз даних [27].

Інтегроване середовище розробки Visual Studio пропонує ряд високорівневих функціональних можливостей, які виходять за рамки базового управління кодом. Нижче перераховані основні особливості середовища для розробки Microsoft Visual Studio [27].

Для обслуговування вебзастосунків ASP.NET необхідний вебсервер, який буде очікувати запити і обробляти відповідні сторінки. Наявність в Microsoft Visual Studio інтегрованого вебсервера дозволяє запускати вебсайт прямо з середовища проєктування, а також підвищує безпеку, виключаючи ймовірність отримання доступу до тестового вебсайту з якого-небудь зовнішнього комп'ютера, оскільки тестовий сервер може приймати з'єднання лише з локального комп'ютера [27].

Для створення більшості додатків потрібно велику кількість стандартного стереотипного коду, і сторінки на ASP.NET тому не виключення. Наприклад, додавання елемента управління, приєднання обробників подій і коригування форматування вимагає установки в розмітці сторінки ряду деталей. У Microsoft Visual Studio такі деталі встановлюються автоматично [27].

За замовчуванням Microsoft Visual Studio форматує код у міру його введення, автоматично вставляючи необхідні відступи і застосовуючи колірне кодування для виділення елементів типу коментарів. Такі незначні відмінності роблять код більш зручним для читання і менш схильним до помилок. Застосовувані Microsoft Visual Studio автоматично параметри форматування можна навіть налаштовувати, що дуже зручно у випадках, коли розробник вважає за краще інший стиль розміщення дужок [27].

Microsoft Visual Studio також має і безліч інших функцій:

– можливість управління проєктом;

– вбудована функція управління вихідним кодом;

– можливість рефакторізації коду;

– потужна модель розширюваності [28].

Більш того, в разі використання Visual Studio Team System розробник отримує розширені можливості для модульного тестування, спільної роботи і управління версіями коду (що значно більше того, що пропонується в більш простих інструментах на кшталт Visual SourceSafe) [28].

Порівняння середовищ для розробки Microsoft Visual Studio та JetBrains Rider наведено в таблиці 2.4.

Таблиця 2.4 – Порівняння середовищ для розробки

Критерій порівняння	Microsoft Visual Studio	JetBrains Rider
Мультиплатформеність	+	+
Підсвічування синтаксису	+	+
Підтримка GitLab	+	+
Аналіз коду	+	+
Менеджер пакетів NuGet	+	+
Провідник рішень	++	+
Технологія автодоповнення	++	+
Споживає небагато пам'яті	+	–
Швидкість відкриття рішення	++	+
Кодова навігація проєкту	++	+
Підтримка вбудованих баз даних	+	+
Модульне тестування	+	+

Пропоновані в Microsoft Visual Studio інструменти налагодження є найкращим засобом для відстеження загадкових помилок і діагностування дивної поведінки. Розробник може виконувати свій код по рядку за раз,

встановлювати інтелектуальні точки переривання, при бажанні зберігаючи їх для використання в майбутньому, і в будь-який час переглядати поточну інформацію з пам'яті [28].

2.2 Засоби для оформлення зовнішнього вигляду застосунку

Інструментами для оформлення зовнішього вигляду застосунку було обрано мову розмітки гіпертекста HTML, для стилізації елементів CSS та придання застосунку динаміки JS. Також фреймворк Bootstrap для імпорту іконок та адаптивної верстки.

Фронтенд – все, що браузер може читати, виводити на екран або запускати. Тобто це HTML, CSS та JS [29].

HTML кажучи простими словами – це мова розмітки всіх елементів і документів на сторінці, і їх взаємодія в структурі сторінки [29].

CSS – це мова характеристики і стилізації зовнішнього вигляду документа. За допомогою CSS-коду браузер розуміє, як саме необхідно відображати елементи. CSS створює шрифти, кольори, визначає розташування блоків сайту, та інше. Також адаптує один і той же документ в різних стилях, виводить передачу на екран або для читання голосом [29].

Завдання JavaScript є відгукуватися на дії користувача, обробляти натискання клавіш, переміщення курсора, кліки мишкою. JavaScript також дає можливість вводити повідомлення, посилати запити на сервер, а також завантажує дані без перезавантаження сторінки, і так далі [30].

Bootstrap входить в число найпопулярніших фреймворків HTML, CSS і JS. Це безкоштовний набір інструментів для створення сайтів, який включає в себе елементи типографіки, готові форми, кнопки, меню та інші елементи. Основним напрямком фреймворку визнана розробка складних мобільних проєктів. Важливою перевагою Bootstrap є велика спільнота, що в п’ять разів перевершує конкурентів [31].

Головна особливість Bootstrap – що це не тільки CSS фреймворк, але і JS-бібліотека. У Bootstrap розроблені готові до використання стилі і скрипти, підключення яких до документа відбувається прописуванням потрібних класів і атрибутів HTML-елементів [31].

Bootstrap визнаний дуже корисним інструментом для мобільної верстки через розробленою у ньому сітки елементів, яка досить гнучка, щоб якісно відображати сайт на екранах найрізноманітніших діагоналей [31].

Головні інструменти фреймворка:

– шаблони – шаблони документа, сторінок;

– сітки – так звані колонки (в Bootstrap їх 12), які є заданими з певними розмірами наперед;

– сповіщення (алерт) – оформлення підказок, діалогових чи спливаючих вікон;

– медіа – спосіб керування зображеннями та відео;

– таблиці – інструмент для оформлення таблиць;

– типографіка – засіб для визначення та оформлення класів для шрифтів, їх опис;

– форми – використовуються для оформлення форм та подій;

– навігація – основне призначення – оформлення сторінок, табів, вкладок тощо [32].

Twitter Bootstrap 3 має ряд своїх переваг, зокрема:

– для роботи з цією версією не обов’язково мати особливі знання у сфері CSS, HTML, JS;

– хороший та стильний дизайн, так як усі класи та компоненти гармоніюють між собою;

– безкоштовність використання;

– сумісність із багатьма браузерами [32].

Переваги Bootstrap:

– наявність великої кількості шаблонів;

– легкість у налаштуванні та використанні;

– економія часу;

– постійне оновлення;

– можливість взаємодії з іншими фреймворками;

– безкоштовне завантаження [32].

2.3 Проєктування вимог до застосунку

На основі проведеного аналізу предметної області було виокремлено нижче приведені вимоги до функціональності застосунку. Основними користувачами виступають кандидати, рекрутери та адміністратори.

Для кандидата:

– аутентифікація облікового запису;

– перегляд та заповнення заявки на вакансію.

Для рекрутера:

– аутентифікація облікового запису;

– перегляд активних вакансій;

– додавання, редагування та видалення вакансій;

– перегляд, створення та заповнення заявок на вакансію;

– перегляд бази замовників;

– додавання та видалення замовників;

– перегляд бази кандидатів;

– додавання, редагування та видалення кандидатів;

– перегляд бази користувачів.

Для адміністратора:

– узгодження або відхилення заявки на вакансію;

– додавання, редагування та видалення користувачів.

Діаграму роботи прецедентів із вебзастосунком для автоматизації завдань рекрутингу зображено на рисунку 2.1.

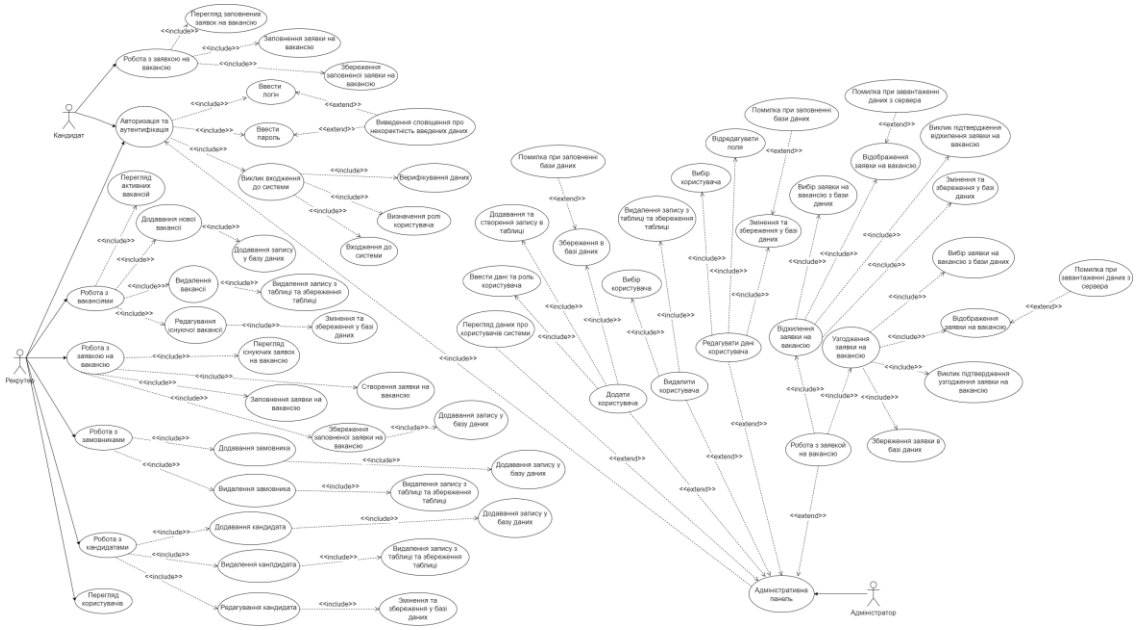

Рисунок 2.1 – Діаграма прецедентів

2.4 Проєктування архітектури застосунку

Архітектурним шаблоном було обрано MVC. Концепція архітектурного шаблона MVC передбачає поділ застосунку на три частини:

– модель (model);

– подання (view);

– контролер (controller) [33].

На рисунку 2.2 зображена схема архітектурного паттерну MVC.

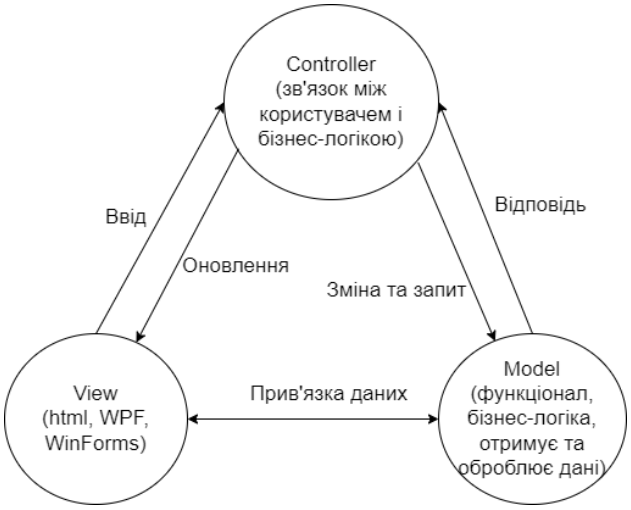

Рисунок 2.2 – Архітектурний паттерн MVC

У цьому шаблоні, призначені для користувача запити направляються в контролер. Контролер викликає модель для виконання дій користувача або отримання даних. Потім контролер передає цю модель в уявлення, і вона повертається користувачеві [33].

Модель в додатку MVC являє стан програми та будь-яку бізнес-логіку або операції, які повинні бути виконані їм. Модель також може містити логіку для збереження стану програми. Уявлення відповідають за подання контенту через призначений для користувача інтерфейс. В ідеалі подання має містити мінімальну логіку, і воно повинно бути пов'язане тільки з поданням контенту [33].

Контролери – це компоненти, які обробляють взаємодію з користувачем, працюють з моделлю і, в кінцевому рахунку, вибирають уявлення для візуалізації. У шаблоні MVC контролер є початковою точкою входу і відповідає за вибір типів моделей для роботи та подання для візуалізації. Іншими словами, контролер контролює, як додаток відповідає на цей запит. Таким чином перевага цього шаблону полягає в тому, що кожен з цих компонентів несе одну відповідальність, і їх простіше кодувати, налагоджувати і тестувати окремо [34].

Серед переваг архітектурного шаблону MVC можна виокремити наступне:

– має архітектуру для надання декількох переглядів;

– допомагає розробити додаток, який завантажується надзвичайно швидко;

– модифікація інтерфейсу користувача не впливає на бізнес-логіку;

– допомагає розробляти більші програми з певною структурою;

– дозволяє повторно використовувати код;

– абстрагування логіки від подання;

– може бути інтегрований з JS [35].

Архітектура вебзастосунку на основі предметної області зображено на рисунку 2.3.

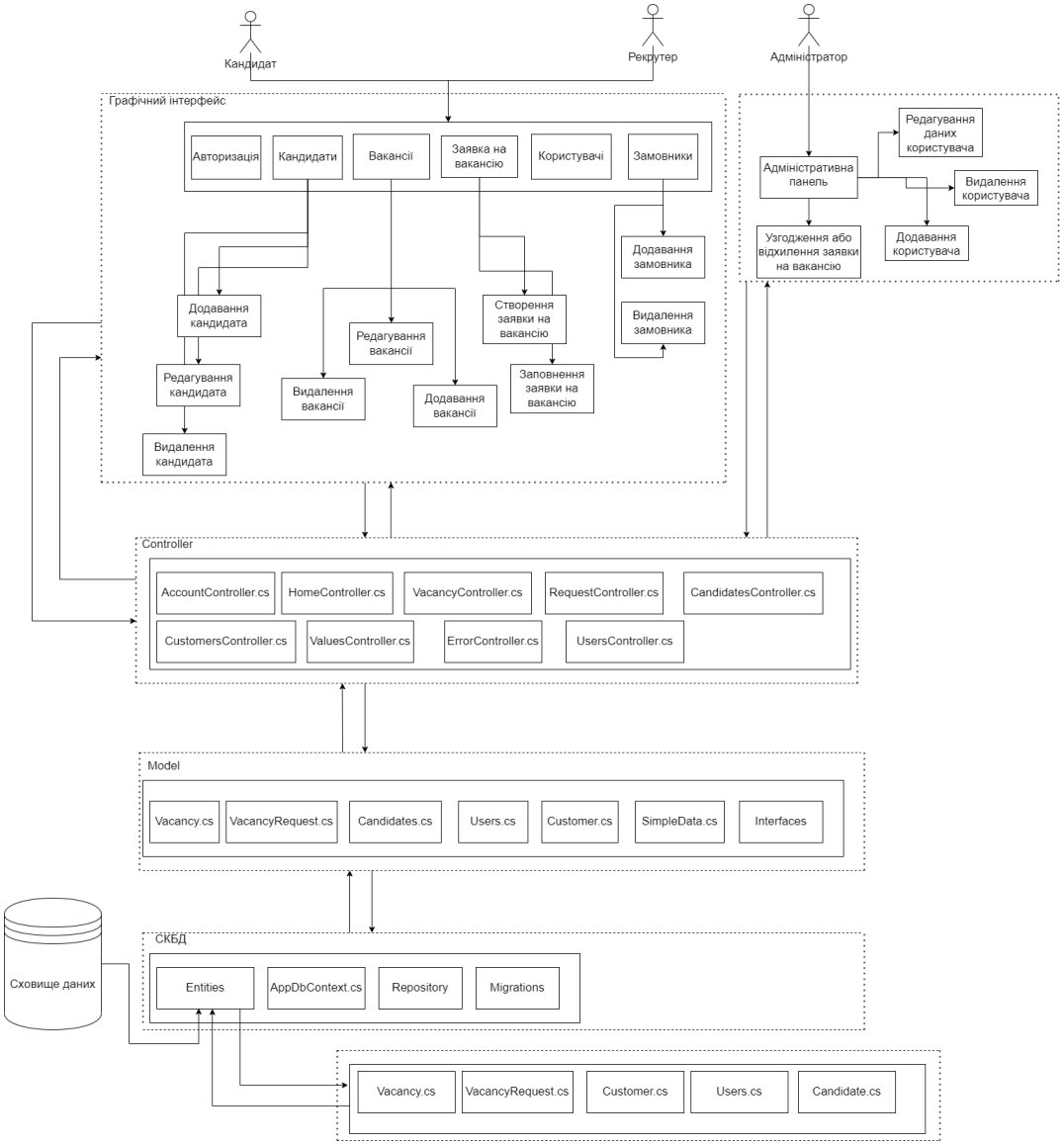

Рисунок 2.3 – Архітектура вебзастосунку

2.5 Проєктування структури бази даних

Для збереження даних вебзастосунок використовує базу даних. Проаналізувавши предметну область було виокремлено наступні п'ять сутностей:

– Vacancy – вакансії;

– VacancyRequest – заявка на вакансію;

– Customer – замовник;

– Users – користувачі;

– Candidate – кандидат.

Нижче буде наведено опис атрибутів усіх сутностей.

У таблиці 2.5 наведено назви атрибутів, типи даних та опис атрибутів сутності Vacancy.

Таблиця 2.5 – Опис сутності Vacancy

№	Назва атрибуту	Тип даних	Опис атрибута
1	Id	int	номер вакансії
2	Name	int	назва
3	Location	nvchar	локація
4	Customer	nvchar	замовник
5	Salary	int	зарплата
6	Open_date	datetime2	дата відкриття
7	Deadline	datetime2	кінцевий термін
8	Responsible_recruiter	nvchar	відповідальний рекрутер
9	Prioirty	nvchar	пріоритет
10	Status	nvchar	статус

У таблиці 2.6 наведено назви атрибутів, типи даних та опис атрибутів сутності VacancyRequest.

Таблиця 2.6 – Опис сутності VacancyRequest

№	Назва атрибуту	Тип даних	Опис атрибута
1	Id	int	номер
2	Vacancy_name	nvchar	назва вакансії
3	Location	nvchar	локація
4	Customer	nvchar	замовник
5	Contact_person	nvchar	контактна особа
6	Creation_date	datetime2	дата створення
7	Deadline	datetime2	кінцевий термін
8	Required_skills	float	необхідні навички

У таблиці 2.7 наведено назви атрибутів, типи даних та опис атрибутів сутності Customer.

Таблиця 2.7 – Опис сутності Customer

№	Назва атрибуту	Тип даних	Опис атрибута
1	Id	int	номер
2	Name	nvchar	назва
3	Status	nvchar	статус
4	Vacancies	float	відкриті вакансії
5	Responsible_persons	float	відповідальні особи
6	Contacts	float	контакти

У таблиці 2.8 наведено назви атрибутів, типи даних та опис атрибутів сутності Users.

Таблиця 2.8 – Опис сутності Users

№	Назва атрибуту	Тип даних	Опис атрибута
1	Id	int	номер
2	Name	nvchar	ПІБ
3	Role	nvchar	роль
4	Region	nvchar	регіон
5	Registration_date	datetime2	дата реєстрації

У таблиці 2.9 наведено назви атрибутів, типи даних та опис атрибутів сутності Candidate.

Таблиця 2.9 – Опис сутності Candidate

№	Назва атрибуту	Тип даних	Опис атрибута
1	Id	int	номер
2	Name	nvchar	ПІБ
3	Position	nvchar	посада
4	Age	int	вік
5	Country	nvchar	країна
6	Work_experience	int	досвід роботи
7	Desired_salary	int	бажана зарплата
8	Source_candidate	float	джерело додавання кандидата

Концептуальна модель бази даних зображена на рисунку 2.4.

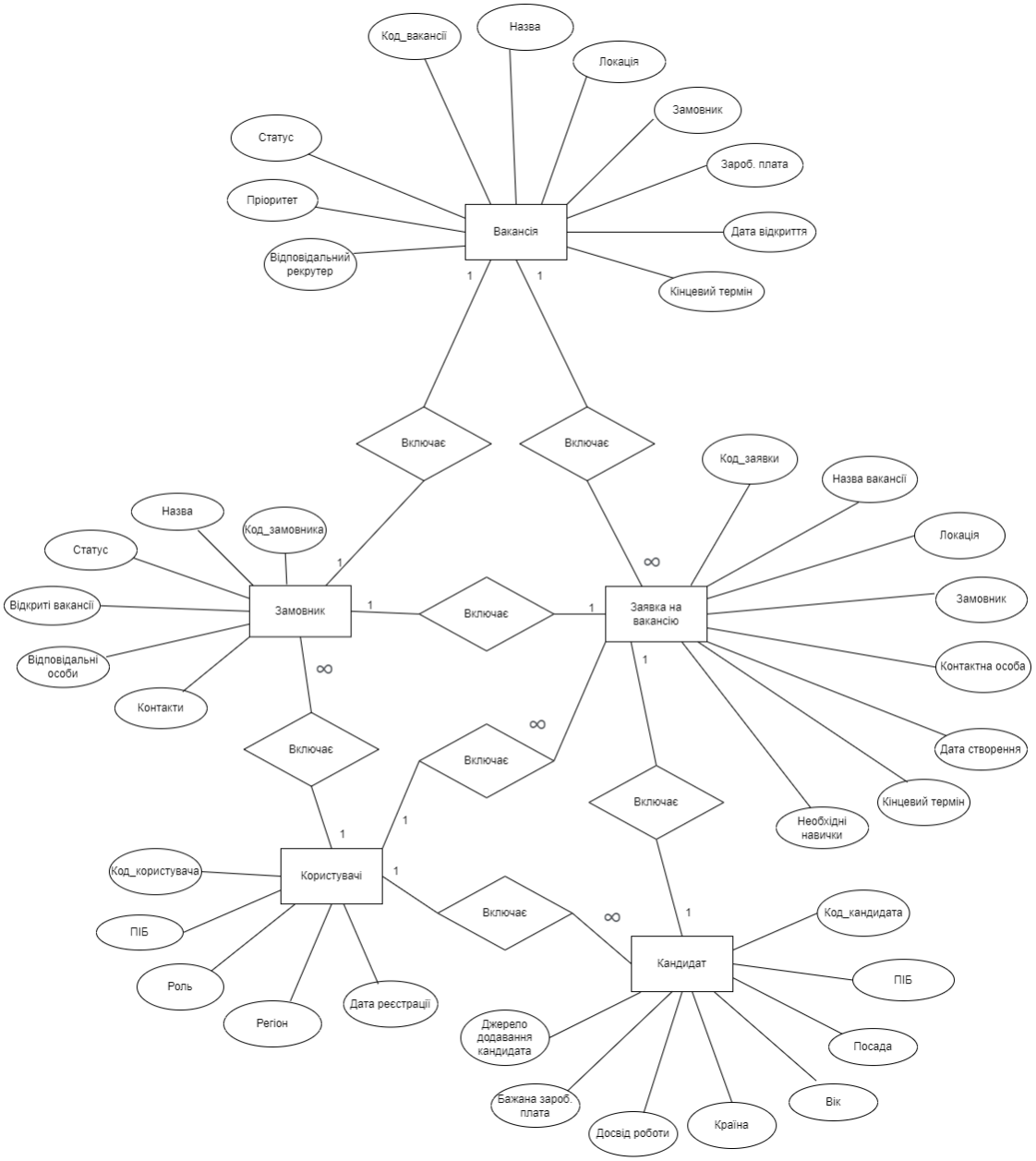

Рисунок 2.4 – Концептуальна модель бази даних

Завдяки такій структурі таблиць зручно зберігати дані про вакансії, кандидатів, користувачів, заявок на вакансію, замовників. Структурну схему бази даних, яка побудована за допомогою СКБД Microsoft SQL Server зображено на рисунку 2.5.

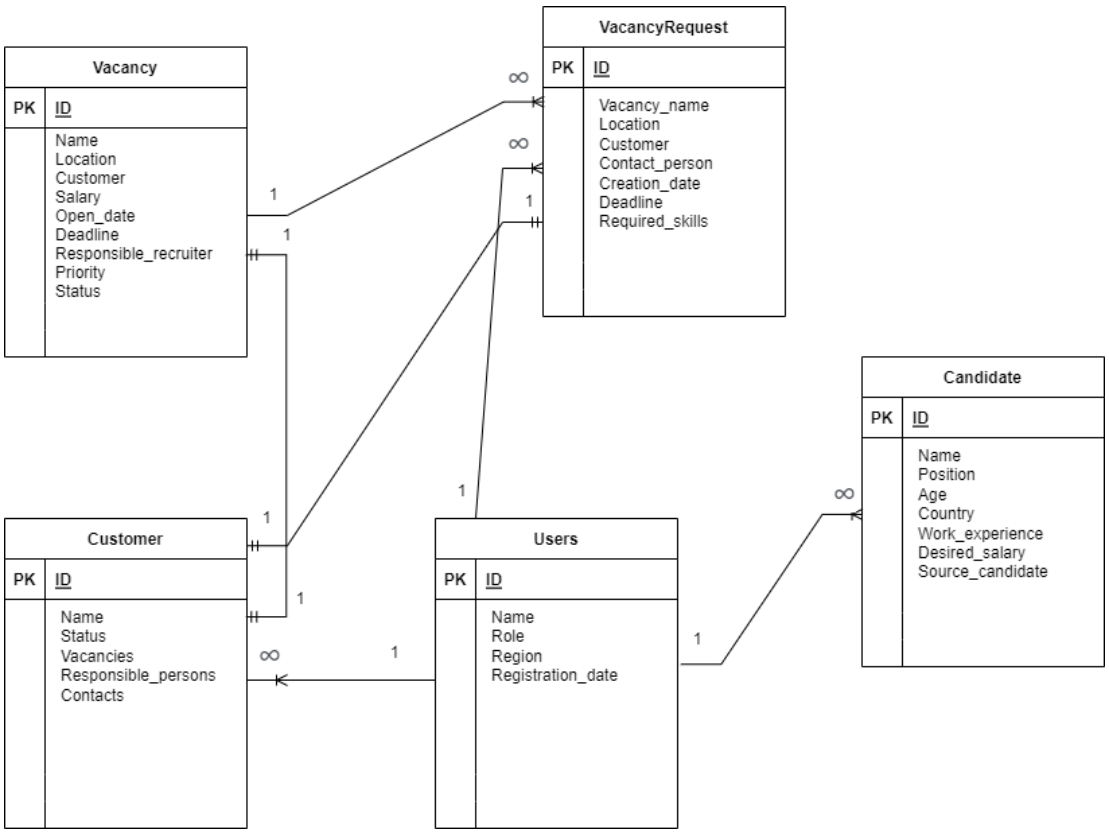

Рисунок 2.5 – Схема структури бази даних

2.6 Схема структури проєкту застосунку

Проєкт було поділено на дві частини. Перша частина відповідає за базу даних, а інша за сам сайт.

Проєкт містить папку Entities у якій зберігаються сутності та їх обмеження по розміру. Також присутні папки у яких зберігаються контролери, моделі та подання застосунку.

Папка Models містить усі дані, пов’язані з моделями для програми, модель – це не що інше, як властивості для зберігання даних, які можна використовувати в програмі для призначення даних та прив’язування їх до представлення. На основі функціональних вимог створено кілька моделей [35].

У папці Controller містяться класи контролерів. Класи контролерів можна розмістити куди завгодно [35].

Папка Views містить усі уявлення для програми [35].

Нижче на рисунку 2.6 можна побачити файлову архітектуру проєкту, яка використовує архітектурний шаблон MVC.

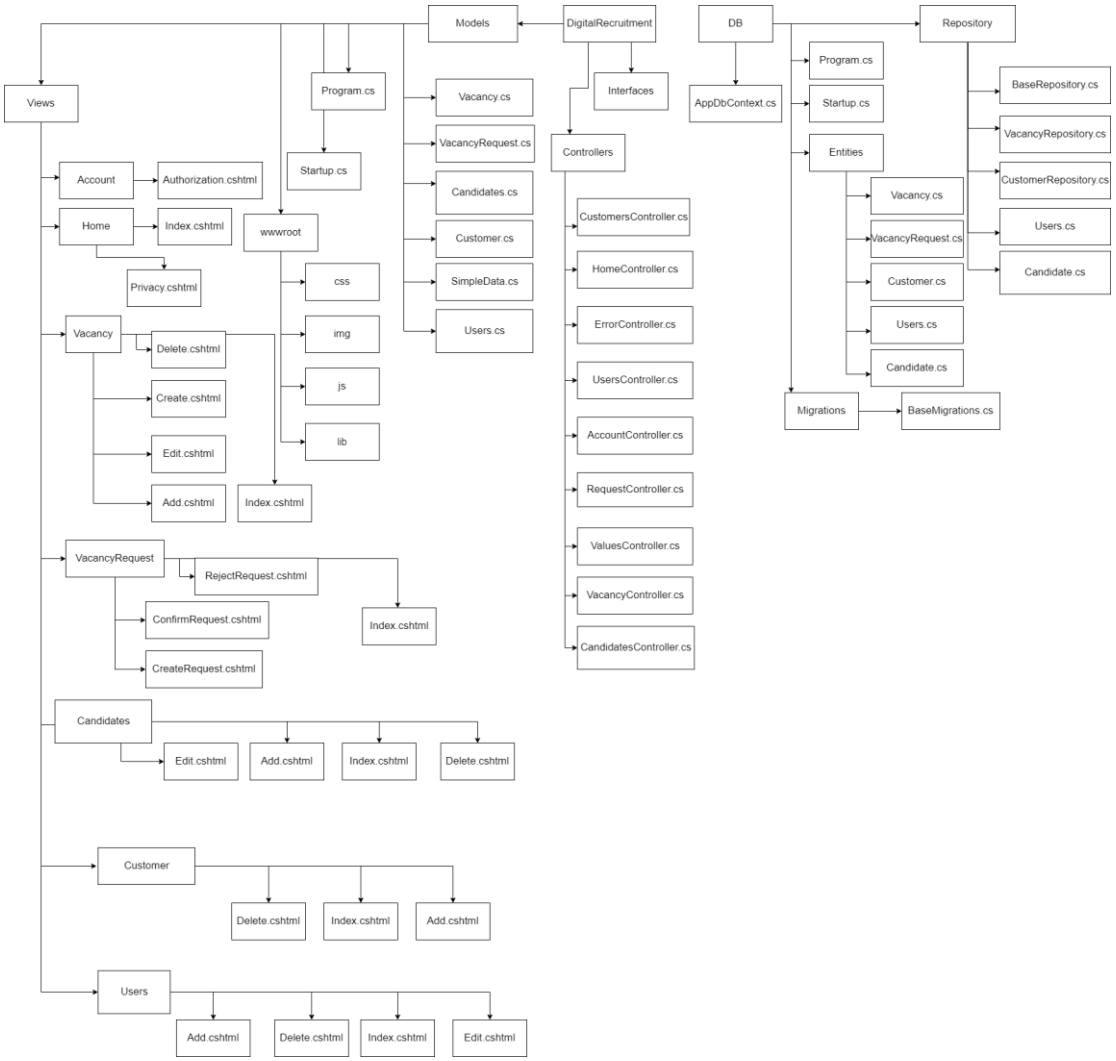

Рисунок 2.6 – Файлова архітектура проєкту

2.7 Висновки за розділом 2

Оглянувши та порівнявши обрані засоби для розробки вебзастосунку з іншими засобами, було обрано використовувати наступні інструменти розробки:

– мова програмування C#;

– фреймворк ASP.NET Core;

– фреймворк для роботи з даними Entity Framework Core;

– середовище для розробки Microsoft Visual Studio;

– мову гіпертексту HTML;

– стилізації елементів сторінок CSS;

– мову програмування JS;

– фреймворк розробки адаптивних застосунків Bootstrap.

У пунктах вище було розглянуто особливості мови програмування C#, фреймворку розробки вебзастосунків ASP.NET Core, фреймворку для роботи з даними Entity Framework Core, СКБД Microsoft SQL Server та середовища для розробки Microsoft Visual Studio.

Критеріями вибору СКБД Microsoft SQL Server є наступні переваги:

– високий ступінь захисту даних;

– потужні засоби роботи з даними;

– висока продуктивність;

– зберігання даних, що вимагають дотримання режиму секретності;

– зберігання великих масивів даних [23].

Також в цьому розділі були приведені вимоги до функціональності застосунку, архітектура вебзастосунку за шаблоном MVC, схему файлової структури проєкту та діаграму роботи прецедентів із застосунком.

Виділено п'ять сутностей та на основі цього проведено проєктування структури бази даних та концептуальне проєктування бази даних.

3 РОЗРОБКА ПРОГРАМНОГО ПРОДУКТУ

3.1 Реалізовані алгоритми функціонування програми

Під час розробки вебзастосунку «DigitalRecruitment» були розроблені наступні алгоритми:

– алгоритм авторизації та аутентифікації облікового запису;

– алгоритм побудови даних про користувачів;

– алгоритм додавання нового користувача;

– алгоритм видалення існуючого користувача;

– алгоритм редагування даних існуючого користувача;

– алгоритм відхилення заявки на вакансію;

– алгоритм узгодження заявки на вакансію;

– алгоритм побудови даних про активні вакансії;

– алгоритм додавання нової вакансії;

– алгоритм редагування існуючої вакансії;

– алгоритм видалення вакансії;

– алгоритм побудови даних про існуючі заявки на вакансію;

– алгоритм створення заявки на вакансію;

– алгоритм заповнення заявки на вакансію;

– алгоритм побудови даних про замовників;

– алгоритм додавання замовників;

– алгоритм видалення замовників;

– алгоритм побудови даних про кандидатів;

– алгоритм додавання кандидатів;

– алгоритм редагування кандидатів;

– алгоритм видалення кандидатів;

– алгоритм заповнення існуючої заявки на вакансію;

– алгоритм побудови даних про заповнені заявки на вакансію.

Таким чином, було реалізовано 23 алгоритма. Ці алгоритми дозволяють забезпечити повноцінне функціонування вебзастосунку.

При запуску застосунку відображається сторінка авторизації, де користувачу необхідно вказати логін та пароль, після натискання на кнопку увійти, здійснюється пошук в базі даних, якщо даного користувача не було знайдено, буде виведено сповіщення про некоректність введених даних.

При самому першому виконанні програми, в базі даних знаходиться лише один користувач (адміністратор), який повинен увійти до системи та додати інших користувачів. Користувач входить до системи, відповідно до його ролі.

Дані про користувачів надходять з бази даних, та формуються у вигляді таблиці, які мають такі поля: номер, ПІБ, роль, регіон, дата реєстрації.

Адміністратор може додати користувача, після натискання на кнопку додати запис, створиться новий порожній запис в таблиці. Адміністратор повинен вказати номер, ПІБ, роль, регіон, дата реєстрації, та після натискання на кнопку зберегти запис, визначається новий ключ для запису та новий запис додається та зберігається в базі даних.

Адміністратор може редагувати дані наявних користувачів, після обрання бажаного запису, адміністратор вносить зміни, та натискає кнопку збереження даних. Після чого починається пошук в базі даних користувача системи, після того як запис було знайдено, визначається поле, яке було відредаговане та замінюється новими даними, які вказав адміністратор та запис зберігається в базі даних.

Адміністратор може видаляти записи вакансій, замовників та кандидатів, після обрання бажаного запису, адміністратор натискає на кнопку видалення запису, після чого запис видаляється з таблиці. Після збереження даних, починається пошук запису в базі даних, після знаходження, запис видаляється з бази даних та дані вакансій, замовників та кандидатів системи зберігаються вже без видаленого запису.

Рекрутер створює заявку для заповнення кандидатом, після цього кандидат заповнює заявку на вакансію, потім адміністратор затверджує або відхиляє.

Дані про вакансії надходять з бази даних, та формуються у вигляді таблиці, які мають такі поля: назва, локація, замовник, заробітна плата, відповідальний рекрутер, пріоритет, статус, дата відкриття вакансії та кінцевий термін.

Дані про кандидатів надходять з бази даних, та формуються у вигляді таблиці, які мають такі поля: ПІБ, посада, вік, країна, досвід роботи, бажана заробітна плата, джерело додавання кандидата.

Дані про існуючі заявки на вакансію надходять з бази даних, та формуються у вигляді таблиці, які мають такі поля: назва вакансії, локація, замовник, контактна особа, дата створення, кінцевий термін, необхідні навички.

Дані про замовників надходять з бази даних, та формуються у вигляді таблиці, які мають такі поля: назва, статус, відкриті вакансії, відповідальні особи, контакти.

Рекрутер може редагувати вакансії та кандидатів, після обрання бажаного запису, рекрутер вносить зміни, та натискає кнопку збереження даних. Після чого починається пошук в базі даних запису вакансії або кандидата, після того як запис було знайдено, визначається поле, яке було відредаговане та замінюється новими даними, які вказав рекрутер та запис зберігається в базі даних.

Рекрутер може видаляти записи вакансій та кандидатів, після обрання бажаного запису, рекрутер натискає на кнопку видалення запису, після чого запис видаляється з таблиці. Після збереження даних, починається пошук запису в базі даних, після знаходження, запис видаляється з бази даних та всі дані вакансій та кандидатів вже без видаленого запису.

Рекрутер може додати вакансію, замовників та кандидатів, після натискання на кнопку додати запис, створиться новий порожній запис в

таблиці. Після натискання на кнопку зберегти запис, визначається новий ключ для запису та новий запис додається та зберігається в базі даних.

Основний алгоритм функціонування вебзастосунку, який виконується при роботі із застосунком зображено на рисунку 3.1.

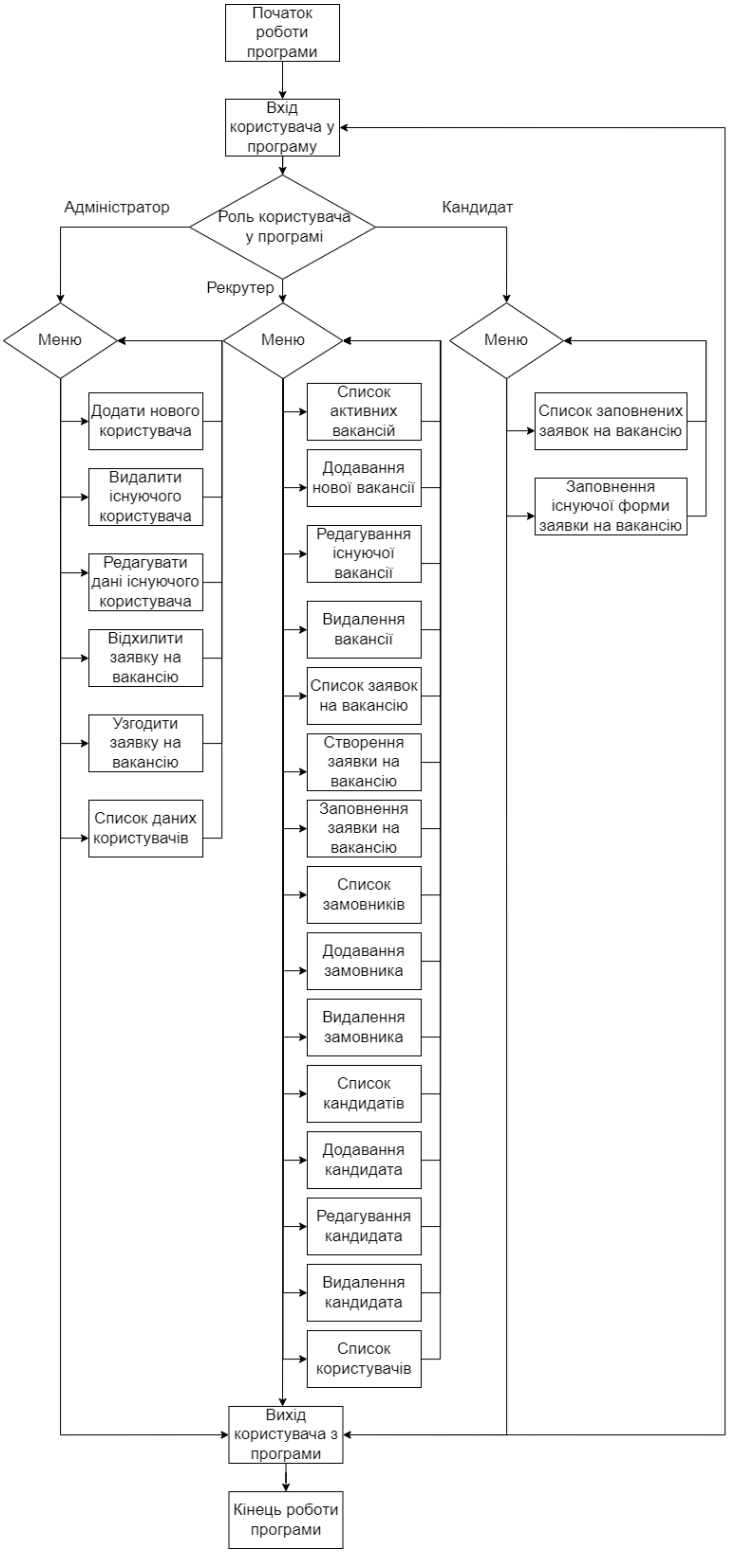

Рисунок 3.1 – Алгоритм функціонування вебзастосунку

3.2 Опис програмних модулів реалізованого застосунку

Програма скаладається з 23 модуля:

– модуль авторизації та аутентифікації облікового запису;

– модуль перегляду заявок на вакансію;

– модуль відображення списку вакансій;

– модуль відображення списку замовників;

– модуль відхилення заявки на вакансію;

– модуль узгодження заявки на вакансію;

– модуль створення заявок на вакансію;

– модуль додавання вакансії;

– модуль збереження вакансій;

– модуль видалення вакансії;

– модуль редагування вакансії;

– модуль додавання замовників;

– модуль видалення замовників;

– модуль відображення списку кандидатів;

– модуль редагування кандидата;

– модуль додавання кандидата;

– модуль збереження кандидата;

– модуль видалення кандидата;

– модуль відображення списку користувачів;

– модуль додавання користувача;

– модуль збереження користувача;

– модуль видалення користувача;

– модуль редагування користувача.

Схема структури модулів реалізованого вебзастосунку зображено на рисунку 3.2.

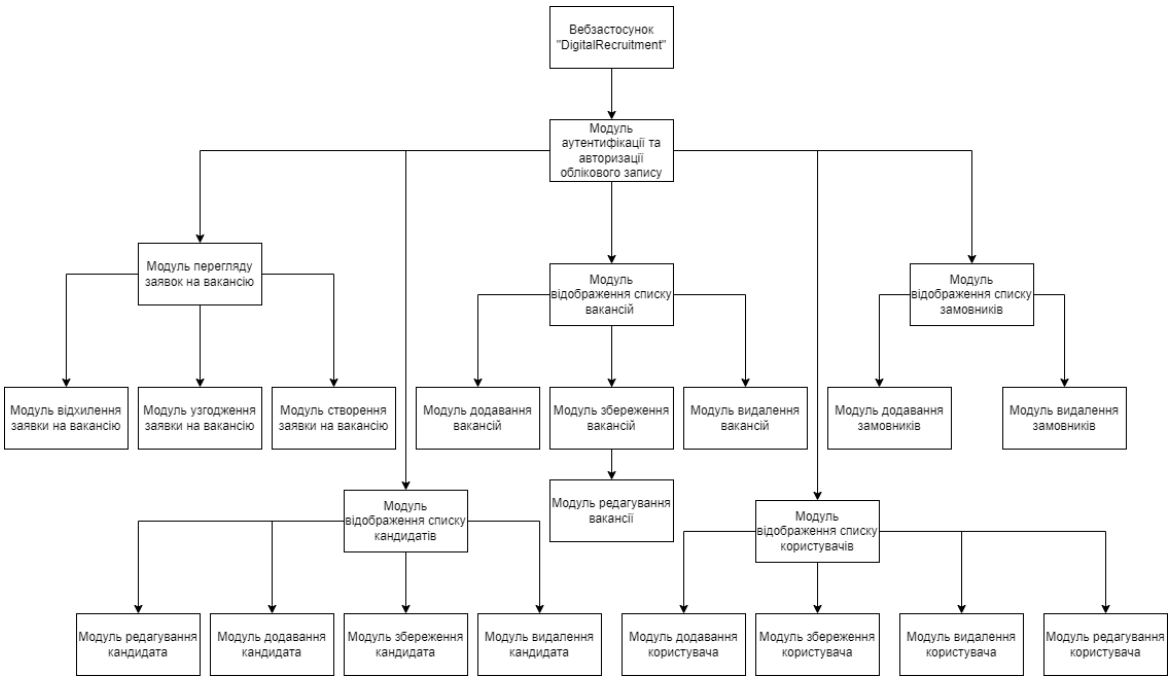

Рисунок 3.2 – Модулі реалізованого вебзастосунку

Детальний опис реалізованих модулів наведено у таблиці 3.1.

Таблиця 3.1 – Опис модулів реалізованого вебзастосунку

Модуль	Опис модулю
Модуль авторизації та аутентифікації облікового запису	У цьому модулі користувач вводить логін та пароль, після цього переходить на сторінку із визначеною роллю та функціональністю.
Модуль перегляду заявок на вакансію	Перегляд існуючих заявок на вакансію.
Модуль відображення списку вакансій	Цей модуль відображає дані вакансії у вигляді таблиці.
Модуль відображення списку замовників	Цей модуль відображає дані замовників у вигляді таблиці.
Модуль відхилення заявки на вакансію	У цьому модулі адміністратор відхиляє заявку на вакансію.
Модуль узгодження заявки на вакансію	У цьому модулі адміністратор узгоджує заявку на вакансію.
Модуль створення заявок на вакансію	Цей модуль відповідає за створення заявки на вакансію.
Модуль додавання вакансії	У цьому модулі рекрутер додає нову вакансію.

Продовження таблиці 3.1

Модуль	Опис модулю
Модуль видалення вакансії	У цьому модулі рекрутер видаляє дані про існуючу вакансію.
Модуль редагування вакансії	У цьому модулі рекрутер редагує дані вакансії.
Модуль додавання замовників	У цьому модулі рекрутер додає нових замовників.
Модуль видалення замовників	У цьому модулі рекрутер видаляє замовників.
Модуль відображення списку кандидатів	Цей модуль відображає дані кандидатів у вигляді таблиці.
Модуль редагування кандидата	У цьому модулі рекрутер редагує дані кандидата.
Модуль додавання кандидата	У цьому модулі рекрутер додає нового кандидата.
Модуль збереження кандидата	У цьому модулі рекрутер зберігає дані кандидата.
Модуль видалення кандидата	У цьому модулі рекрутер видаляє дані існуючого кандидата.
Модуль відображення списку користувачів	Цей модуль відображає дані користувачів у вигляді таблиці.
Модуль додавання користувача	У цьому модулі адміністратор додає нового користувача.
Модуль збереження користувача	У цьому модулі адміністратор зберігає дані користувача.
Модуль видалення користувача	У цьому модулі адміністратор видаляє дані існуючого користувача із бази даних.
Модуль редагування користувача	У цьому модулі адміністратор редагує дані існуючого користувача.

3.3 Опис реалізованих функцій застосунку

Вебзастосунок складається з необхідних функцій, які вирішують усі поставлені задачі для реалізації програми.

Були використані стандартні та асинхронні функції для отримання та передання оновлених даних з сервера.

Нижче у таблиці 3.2 наведені функції, які виконуються на стороні сервера.

Таблиця 3.2 – Функції серверної частини застосунку

Функція	Опис функції
Initialize()	Ініціалізує базу даних при запуску вхідними даними.
Create()	Створює запис у базі даних.
Update()	Оновлює наявний запис у базі даних.
Add()	Додає запис у базу даних.
Get()	Отримує дані з серверу.
GetVacancyAsyns()	Отримує дані про вакансії з серверу асинхронним методом.
GetRequestAsyns()	Отримує дані про заявки на вакансію з серверу асинхронним методом.
SaveChages()	Зберігає в базі даних змінені дані.
GetUsersAsyns()	Отримує дані про користувачів з серверу асинхронним методом.
GetCustomerAsyns()	Отримує дані про замовників з серверу асинхронним методом.
GetCandidateAsyns()	Отримує дані про кандидатів з серверу асинхронним методом.

3.4 Висновки за розділом 3

У даному розділі наводяться реалізовані алгоритми функціонування програми та сама функціональна схема розробленого застосунку, яка показує як виконується робота користувача в залежності від наданих йому прав. Також у даному розділі описано кожний розроблений модуль та приведена схема реалізованих модулів застосунку.

4 ЕКСПЛУАТАЦІЯ ТА ТЕСТУВАННЯ ПРОГРАМНОГО ПРОДУКТУ

4.1 Умови експлуатації застосунку та призначення програми

Програмний продукт призначений для автоматизації завдань рекрутингу. Застосунок може використовуватися для користувачів, яким необхідно зручно та швидко внести до бази вакансії, кандидатів, замовників та додати заявку на вакансію.

Застосунок має такі функціональні можливості:

– аутентифікація облікового запису;

– перегляд активних вакансій;

– додавання, редагування та видалення вакансій;

– перегляд, створення та заповнення заявок на вакансію;

– узгодження або відхилення заявки на вакансію;

– перегляд бази замовників;

– додавання та видалення замовників;

– перегляд бази кандидатів;

– додавання, редагування та видалення кандидатів;

– перегляд бази користувачів;

– додавання, редагування та видалення користувачів.

До апаратної частини сервера повинні виконуватися наступні умови:

– процесор не нижче AMD Ryzen або Intel i5;

– оперативна пам’ять не менше ніж 2 гб;

– жорсткий диск на 1 тб;

– мережева карта.

До апаратної частини клієнту повинні виконуватися наступні умови:

– процесор AMD FX-6100 або Intel i5 чи вище;

– оперативна пам’ять не менше ніж 1 гб;

– жорсткий диск на 500 гб;

– мережева карта;

– комп’ютерна миша та клавіатура.

4.2 Характеристики програми

Назва вебзастосунку – «DigitalRecruitment».

Програма розташовується на жорсткому диску.

Список файлів, з яких складається програма:

– Program.cs – файл в якому визначена точка входу в застосунок метод Main;

– SimpleData.cs – інформація для створення таблиць і з подальшим заповненнями даними;

– ValuesController.cs – REST API методи, що визначають бізнес-логіку програми;

– StartUp.cs – визначає поведінку і додаткові можливості вебпроєкту, а так само конфігурацію з'єднання з базою даних;

– Vacancy.cs – визначення класу вакансій, за яким визначається структура таблиці бази даних;

– VacancyRequest.cs – визначення класу заявок на вакансію, за яким визначається структура таблиці бази даних;

– AppDbContext.cs – визначення класу, який використовується для зв'язку таблиці баз даних з класами моделі;

– Customer.cs – визначення класу замовників, за яким визначається структура таблиці бази даних;

– Candidate.cs – визначення класу кандидатів, за яким визначається структура таблиці бази даних;

– Users.cs – визначення класу користувачів, за яким визначається структура таблиці бази даних;

– wwwroot – директорія, що містить, збілдений і мінімізований клієнт вебзастосунку;

– appsettings – основні налаштування вебпроєкту для застосунка ASP.NET Core.

4.3 Виконання програми

При запуску вебзастосунку з'являється форма входу у систему (авторизація), де необхідно ввести логін та пароль (рис. 4.1). Після натискання на кнопку увійти, здійснюється пошук в базі даних, якщо даного користувача не було знайдено, буде виведено сповіщення про некоректність введених даних. Користувачів додає адміністратор.

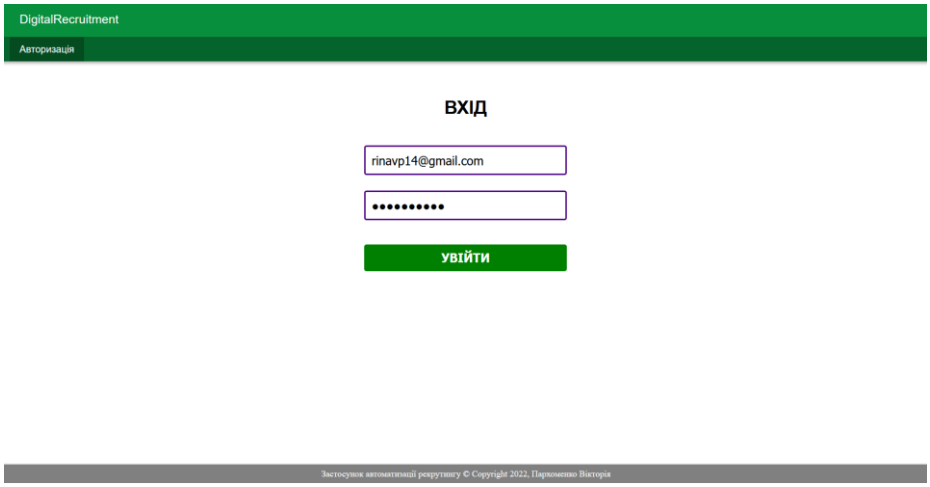

Рисунок 4.1 – Сторінка входу у застосунок

Адміністратор має можливість додавати, редагувати та видаляти дані користувачів (рис. 4.2), також узгоджувати або відхиляти заявку на вакансію.

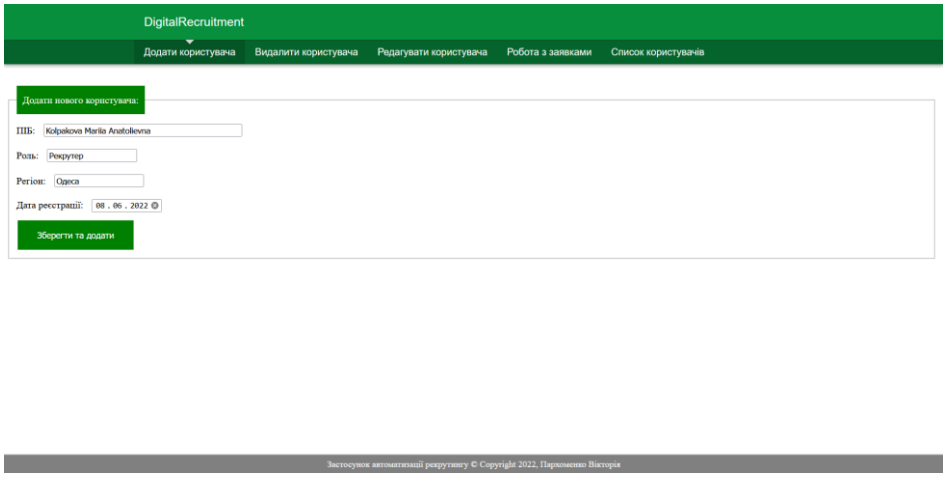

Рисунок 4.2 – Сторінка додавання нового користувача

Кандидат має можливість заповнювати заявку на вакансію, переглядати заявки на вакансію (рис. 4.3).

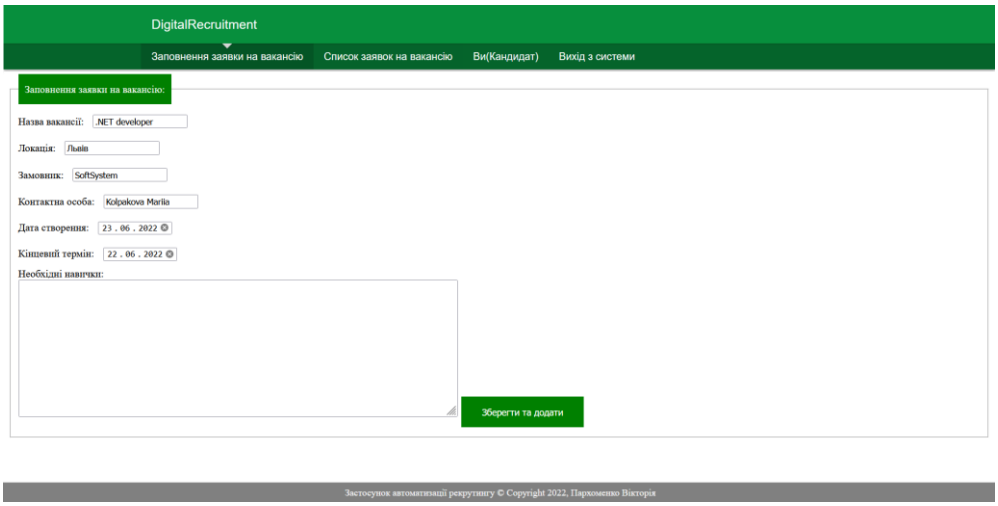

Рисунок 4.3 – Сторінка заповнення заявки на вакансію

Рекрутер має можливість додавати вакансії (рис. 4.4), кандидатів, замовників, створювати та заповнювати заявку на вакансію.

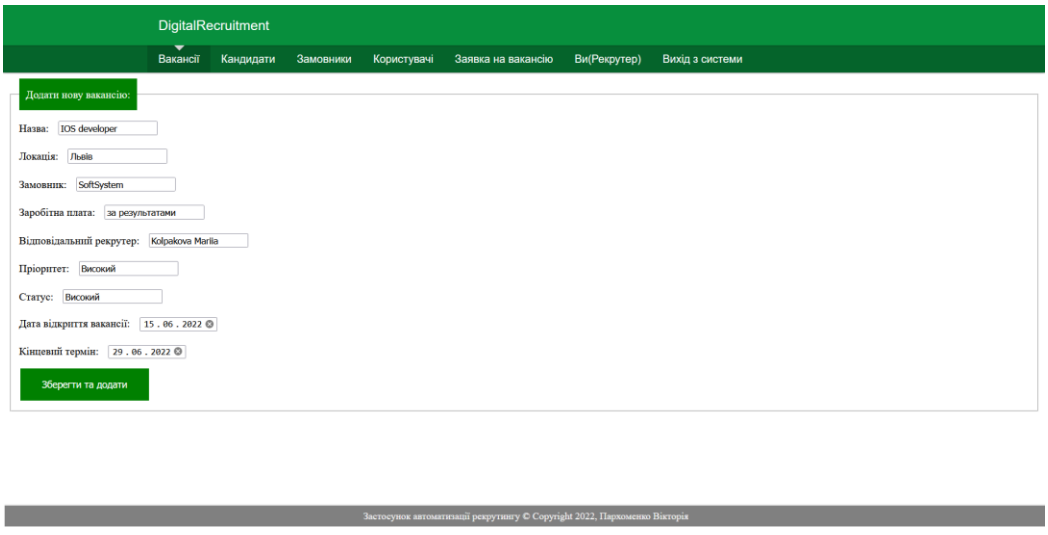

Рисунок 4.4 – Сторінка додавання нової вакансії

4.4 Оповіщення користувачу

Застосунок «DigitalRecruitment» може вивести повідомлення з помилкою "Помилка при завантаженні даних з сервера" (рис. 4.5).

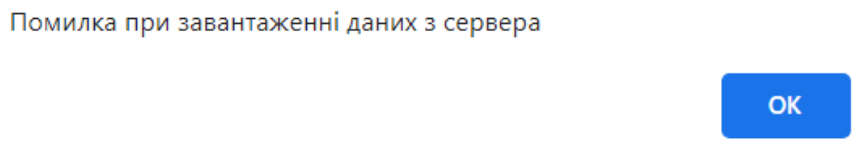

Рисунок 4.5 – Помилка при завантаженні даних з сервера

Застосунок «DigitalRecruitment» може вивести повідомлення з помилкою "Некоректні введені дані логіну або паролю" (рис. 4.6).

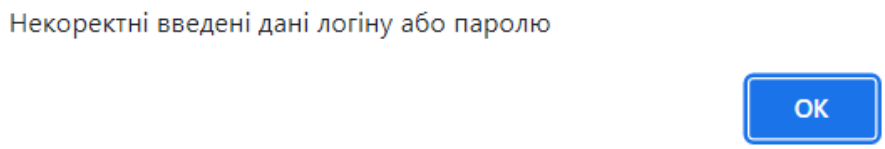

Рисунок 4.6 – Некоректні введені дані логіну або паролю

Застосунок «DigitalRecruitment» може вивести повідомлення з помилкою "Сталася помилка під час заповнення бази даних" (рис. 4.7).

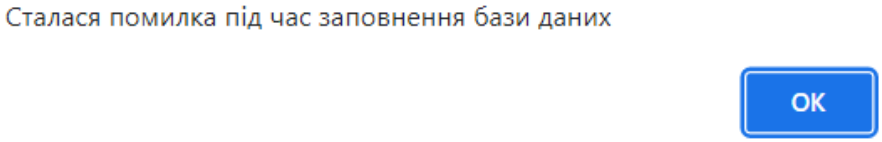

Рисунок 4.7 – Сталася помилка під час заповнення бази даних

Застосунок «DigitalRecruitment» може вивести повідомлення з помилкою для кандидата "Заявка на вакансію відправлена адміністратору на розгляд" (рис. 4.8).

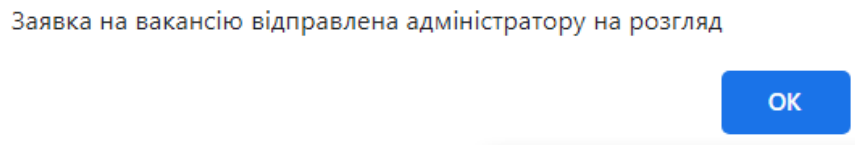

Рисунок 4.8 – Заявка на вакансію відправлена адміністратору на розгляд

4.5 Опис методики тестування та отриманих результатів

Було вирішено створити чек-лист перевірки функціоналу модулів застосунку у браузерах Google Chrome та Opera. Чек-лист перевірки функціоналу модулів наведено у таблиці 4.1.

Таблиця 4.1 – Чек-лист перевірки функціоналу модулів застосунку

Застосунок «DigitalRecruitment»	Google Chrome	Opera
Модуль авторизації та аутентифікації облікового запису	Пройдено	Пройдено
Модуль перегляду заявок на вакансію	Пройдено	Пройдено
Модуль відображення списку вакансій	Пройдено	Пройдено
Модуль відображення списку замовників	Пройдено	Пройдено
Модуль відхилення заявки на вакансію	Пройдено	Пройдено
Модуль узгодження заявки на вакансію	Пройдено	Пройдено
Модуль створення заявок на вакансію	Пройдено	Пройдено
Модуль додавання вакансії	Пройдено	Пройдено
Модуль збереження вакансій	Пройдено	Пройдено
Модуль видалення вакансії	Пройдено	Пройдено
Модуль редагування вакансії	Пройдено	Пройдено
Модуль додавання замовників	Пройдено	Пройдено
Модуль видалення замовників	Пройдено	Пройдено
Модуль відображення списку кандидатів	Пройдено	Пройдено
Модуль редагування кандидата	Пройдено	Пройдено
Модуль додавання кандидата	Пройдено	Пройдено
Модуль збереження кандидата	Пройдено	Пройдено
Модуль видалення кандидата	Пройдено	Пройдено

Продовження таблиці 4.1

Застосунок «DigitalRecruitment»	Google Chrome	Opera
Модуль відображення списку користувачів	Пройдено	Пройдено
Модуль додавання користувача	Пройдено	Пройдено
Модуль збереження користувача	Пройдено	Пройдено
Модуль видалення користувача	Пройдено	Пройдено
Модуль редагування користувача	Пройдено	Пройдено

4.6 Висновки за розділом 4

Даний розділ описує умови екплуатації та призначення вебзастосунку, характеристики застосунку, можливі сповіщення про помилки та приведено чек-лист перевірки функціоналу модулів застосунку.

ВИСНОВКИ

На початку роботи над дипломною роботою було оглянуто аналогічні програмні застосунки та приведено їх порівняння за спільними властивостями. Визначено, якою функціональністю повинен володітити застосунок для автоматизації завдань рекрутингу на основі розгляду предметної області.

Були приведені вимоги до функціональності застосунку, архітектура вебзастосунку за паттерном MVC, схема файлової структури проєкту та діаграма роботи прецедентів із застосунком. Виділені сутності та на основі цього проведено проєктування структури бази даних та концептуальне проєктування бази даних.

Було обрано використовувати мову програмування C#, фреймворк ASP.NET Core, фреймворк для роботи з даними Entity Framework Core, середовище для розробки Microsoft Visual Studio, мову гіпертексту HTML, таблицю стилізації елементів сторінок CSS, мову програмування JS, фреймворк розробки адаптивних застосунків Bootstrap.

Наведено реалізовані алгоритми функціонування програми та саму функціональну схему розробленого застосунку. Також описано кожний розроблений модуль та приведена схема реалізованих модулів застосунку.

У даному вебзастосунку для автоматизації завдань рекрутингу реалізовані наступні функції:

– узгодження або відхилення заявки на вакансію;

– додавання, редагування та видалення користувачів.

– додавання, редагування та видалення вакансій;

– створення та заповнення заявок на вакансію;

– додавання та видалення замовників;

– додавання, редагування та видалення кандидатів.

Результатом тестування вебзастосунка є успішність виконання усіх функції поставлених в процесі тестування.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Тенденції розвитку рекрутингу персоналу через призму диджитал-інновацій [Електрон. ресурс]. – Режим доступу: https://ir.kneu.edu.ua/handle/2010/36513.

2. Рекрутинг персоналу [Текст] / С.О. Цимбалюк. // Київ: КНЕУ, – 2019. – С. 9-15.

3. Використання соціальних мереж у сучасному рекрутингу України [Електрон. ресурс]. – Режим доступу: http://www.prostir.pdaba.dp.ua/index.php/journal/article/view/205.

4. Як розробити ПЗ для підбору персоналу і не потрапити у халепу [Електрон. ресурс]. – Режим доступу: https://cleverstaff.net/blog/uk/how-to-develop-the-recruiting-soft/.

5. Applicant Tracking System, ATS [Електрон. ресурс]. – Режим доступу: https://www.it.ua/knowledge-base/technology-innovation/applicant-tracking-system-ats.

6. Найпопулярніша в Україні система для рекрутингу [Електрон. ресурс]. – Режим доступу: https://cleverstaff.net/ua/.

7. PersiaHR. Онлайн-рішення для автоматизації роботи команди рекрутерів [Електрон. ресурс]. – Режим доступу: https://www.persiahr.com/uk.

8. Як економити на автоматизації рекрутингу в період кризи [Електрон. ресурс]. – Режим доступу: https://hurma.work/blog/yak-ekonomiti-na-avtomatizacziyi-rekrutingu-v-period-krizi/.

9. Автоматизація рекрутингу. Практичний посібник з вибору, впровадження та застосування ATS-систем [Текст] / В.О. Сиротенко // Київ: КНЕУ, – 2022. – С. 10-30.

10. Система для автоматизації рекрутинга «Hurma» [Електрон. ресурс]. – Режим доступу: https://hurma.work/.

11. Вся вирва рекрутингу – як на долоні [Електрон. ресурс]. – Режим доступу: https://hurma.work/ru/capabilities/recruiting/.

12. REST API Architectural Constraints [Electronic resource] / – Access mode: https://www.geeksforgeeks.org/rest-api-architectural-constraints/.

13. C# Language advantages and applications [Electronic resource] / – Access mode: https://www.tutorialspoint.com/Chash-Language-advantages-and-applications.

14. Об'єктно-орієнтоване програмування (Частина 2) [Електрон. ресурс]. – Режим доступу: http://iwanoff.inf.ua/oop_2_ua/LabTraining01.html.

15. Введение в C# [Электрон. ресурс]. – Режим доступа: https://metanit.com/sharp/tutorial/1.1.php.

16. 9 причин вивчити мову C# [Електрон. ресурс]. – Режим доступу: https://foxminded.ua/ua/9-prichin-vivchiti-movu-c/.

17. Чому варто вивчати мову програмування C# (C Sharp) [Електрон. ресурс]. – Режим доступу: https://www.quality-assurance-group.com/chomu-varto-vyvchaty-movu-programuvannya-c-c-sharp/.

18. Розробка ASP.NET [Електрон. ресурс]. – Режим доступу: https://brights.io/ua/services/web-development/asp-net.

19. Введение в ASP.NET Core [Электрон. ресурс]. – Режим доступа: https://metanit.com/sharp/aspnet6/1.1.php.

20. Microsoft SQL Server [Електрон. ресурс]. – Режим доступу: https://tiengtrung.cn/wiki/uk/Microsoft_SQL_Server.

21. Introduction to Database Management Systems (DBMS) | Core CS [Electronic resource] / – Access mode: https://workat.tech/core-cs/tutorial/intro-to-database-management-systems-dbms-8bn0mpx2np3m.

22. Реляційна модель бази даних: елементи, як це зробити, приклад [Електрон. ресурс]. – Режим доступу: https://uk.warbletoncouncil.org/modelo-relacional-base-datos-935.

23. How does a relational database work [Electronic resource] / – Access mode: http://coding-geek.com/how-databases-work/.

24. What is Entity Framework? [Electronic resource] / – Access mode: https://entityframework.net/what-is-ef.

25. Introduction to Entity Framework [Electronic resource] / – Access mode: https://www.dotnettricks.com/learn/entityframework/introduction-to-entity-framework.

26. Введение в Entity Framework [Электрон. ресурс]. – Режим доступа: https://metanit.com/sharp/entityframework/1.1.php.

27. Welcome to the Visual Studio IDE [Electronic resource] / – Access mode: https://docs.microsoft.com/en-us/visualstudio/get-started/visual-studio-ide?view=vs-2022.

28. Visual Studio [Electronic resource] / – Access mode: https://www.incredibuild.com/integrations/visual-studio.

29. Розробка з боку Front end – що це таке і чим відрізняється від Back end? [Електрон. ресурс]. – Режим доступу: https://dan-it.com.ua/uk/blog/rozrobka-z-boku-front-end-shho-ce-take-i-chim-vidriznjaietsja-vid-back-end/.

30. Простыми словами о «фронтенде» и «бэкенде»: что это такое и как они взаимодействуют [Электрон. ресурс]. – Режим доступа: https://tproger.ru/translations/frontend-backend-interaction/.

31. Ознайомлення з Bootstrap [Електрон. ресурс]. – Режим доступу: https://armedsoft.com/ua/blog/oznayomlennya-z-bootstrap.

32. Bootstrap: що це, з чого почати вивчення і як використовувати [Електрон. ресурс]. – Режим доступу: https://druzy.com.ua/bootstrap-sho-ce-z-chogo-pochati-vivchennia-i-iak-vikoristovyvati/.

33. Шаблон проектування MVC [Електрон. ресурс]. – Режим доступу: http://yoip.com.ua/shablon-proektuvannya-mvc/.

34. Общие сведения ASP.NET Core MVC [Электрон. ресурс]. – Режим доступа: https://docs.microsoft.com/ru-ru/aspnet/core/mvc/overview?view=aspnetcore-6.0.

35. Що таке MVC? [Електрон. ресурс]. – Режим доступу: https://uk.education-wiki.com/3886982-what-is-mvc.

ДОДАТОК А

Технічне завдання

Вступ

Застосунок для автоматизації завдань рекрутингу можна застосовувати на будь-якому підприємстві або компанії. Програмний продукт дозволяє автоматизувати та зменшити часові затрати на виконання повсякденних завдань. Основними користувачами будуть виступати кандидати, рекрутери та адміністратори.

Програмний продукт повинен бути реалізований у вигляді вебзастосунку.

А.1 Підстави для розробки

Відповідно до наказу № 102 від 27 квітня 2022 р. затвердженим Національним університетом «Запорізька політехніка» дипломна кваліфікаційна робота виконується за темою «Програмна реалізація застосунку для автоматизації завдань рекрутингу».

А.2 Назва програми

Назва застосунку – «DigitalRecruitment».

А.3 Призначення розроблюваного застосунку

Програмний продукт призначений для автоматизації завдань рекрутингу. Розроблюваний застосунок може використовуватися для користувачів, яким необхідно зручно та швидко внести до бази вакансії, кандидатів, замовників та додати заявку на вакансію.

Уся необхідна інформація повинна зберігатися в базі даних та мати можливість для її перегляду та коригування.

А.4 Вимоги до розроблюваної програми

А.4.1 Функціональні характеристики

Вебзастосунок для автоматизації завдань рекрутингу повинен надавати наступні функціональні можливості в залежності від типу користувача.

Для кандидата:

– аутентифікація облікового запису;

– перегляд та заповнення заявки на вакансію.

Для рекрутера:

– аутентифікація облікового запису;

– перегляд активних вакансій;

– додавання, редагування та видалення вакансій;

– перегляд, створення та заповнення заявок на вакансію;

– перегляд бази замовників;

– додавання та видалення замовників;

– перегляд бази кандидатів;

– додавання, редагування та видалення кандидатів;

– перегляд бази користувачів.

Для адміністратора:

– узгодження або відхилення заявки на вакансію;

– додавання, редагування та видалення користувачів.

Вхідними даними застосунку є наступна інформація:

– при аутентифікації є логін та пароль користувача;

– при додаванні вакансії вказується загальна інформація про вакансію: назва, локація, замовник, заробітна плата, відповідальний рекрутер, пріоритет, статус, дата відкриття вакансії та кінцевий термін;

– при додаванні кандидата вказується загальна інформація про кандидата: ПІБ, посада, вік, країна, досвід роботи, бажана заробітна плата, джерело додавання кандидата;

– при заповненні заявки на вакансію вказується загальна інформація про заявку: назва вакансії, локація, замовник, контактна особа, дата створення, кінцевий термін, необхідні навички;

– при додаванні користувача вказується загальна інформація про користувача: ПІБ, роль, регіон, дата реєстрації;

– при додаванні замовника вказується назва, статус, відкриті вакансії, відповідальні особи, контакти.

Вихідними даними застосунку є наступна інформація:

– затвердження або відхилення заявки на вакансію;

– відображення даних про вакансії, кандидатів, користувачів, замовників, заявок на вакансію.

А.4.2 Вимоги до стабільності функціонування програми

Стабільне функціонування застосунку повинно забезпечуватися наступними факторами, перелік яких наведено нижче:

– організацією безперебійного живлення апаратно-технічних засобів;

– контроль початкових та вихідних даних;

– підключення до мережі Інтернет;

– наявність повідомлень про помилки, які можливі внаслідок некоректних дій користувача при взаємодії з програмою.

А.4.3 Умови експлуатації застосунку

Необхідно використовувати мову програмування C# та середовище для розробки Microsoft Visual Studio 2022. Також це необхідно для розробки серверної частини під ASP.NET Core.

Для клієнта необхідна наявність браузера, наприклад Google Chrome, Microsoft Edge, Mozila Firefox тощо. Також повинна бути встановлена операційна система Windows або Linux для клієнта.

На сервері необхідно, щоб було встановлено СКБД Microsoft SQL Server.

Умовами експлуатації вебзастосунку є необхідність встановлення наступних програмних бібліотек:

– ASP.NET Core MVC версії 2.2.5;

– Microsoft ASP.NET Core Authentication JwtBearer версії 5.0;

– Microsoft ASP.NET Core Identity Entity Framework Core версії 6.0.5;

– Microsoft Entity Framework Core Design версії 6.0.5;

– Microsoft Entity Framework Core Tools версії 6.0.5;

– Microsoft Entity Framework Core версії 6.0.5;

– Microsoft Entity Framework Core SQL Server версії 6.0.5;

– Bootstrap версії 5.1.3;

– AutoMapper версії 11.0.1;

– .NET Core версії 6.0.

Повноцінне функціонування програмного продукту за відсутності даних програмних бібліотек неможливе. Програмні бібліотеки можна встановити через пакетний менеджер NuGet.

А.4.4 Параметри технічних засобів

Для експлуатації застосунку необхідні наступні параметри до технічних засобів:

– процесор не нижче Intel i5 або будь-який сучасний процесор;

– обсяг оперативного запам'ятовуючого пристрою – не менше 4 Гб;

– монітор з роздільною здатністю не менше 1920x1080 або інший засіб виводу зображення;

– обсяг вільного простору на жорсткому диску – не менше 1 Гб;

– відеокарта не менше NVIDIA GeForce GTX 1650;

– комп’ютерна миша та клавіатура.

А.4.5 Маркування програмного продукту

Застосунок може бути записаний на жорсткому диску або флеш-диску. На пристрої або упаковці мають бути такі маркування: «Програмна реалізація застосунку для автоматизації завдань рекрутингу».

А.4.6 Зберігання програмного продукту

Вимоги до збереження не відрізняються від вимог, які були рекомендовані до носія, на якому зберігається застосунок, що включає: жорсткий диск або флеш-диск.

А.5 Етапи розробки

Над дипломною кваліфікаційною роботою бакалавра заплановано виконання наступних основних етапів:

– аналіз предметної області;

– проєктування архітектури та структури програми;

– розробка програмного продукту;

– експлуатація та тестування програмного продукту;

– оформлення відповідної документації.

А.6 Послідовність приймання роботи

Дипломна робота виконується у відповідності з графіком календарного плана. Дипломна робота приймається консультантами відповідних розділів, підписується керівником роботи та нормоконтролером, після чого допускається до захисту завідуючим кафедрою та захищається в присутності членів екзаменаційної комісії. Також необхідно подати роботу рецензенту та отримати рецензію на дипломну роботу.

ДОДАТОК Б

Текст програми

Б.1 Текст класу Vacancy

```
public class Vacancy
    {
        public int Id { get; set; }
        [Required]
        [MaxLenght(15)]
        public string Name { get; set; }
        [Required]
        [MaxLenght(30)]
        public string Location { get; set; }
        [Required]
        [MaxLenght(30)]
        public string Customer { get; set; }
        [Required]
        [MaxLenght(40)]
        public string Salary { get; set; }
        public DateTime Open_date { get; set; }
        public DateTime Deadline { get; set; }
        [Required]
        [MaxLenght(30)]
        public string Responsible_recruiter { get; set; }
        [Required]
        [MaxLenght(30)]
        public string Priorirty { get; set; }
        [Required]
        [MaxLenght(30)]
        public string Status { get; set; }
    }
```

Б.2 Текст класу VacancyRequest

```
public class VacancyRequest
    {
        public int Id { get; set; }
        [Required]
        [MaxLenght(30)]
        public string Vacancy_name { get; set; }
        [Required]
        [MaxLenght(30)]
        public string Location { get; set; }
        [Required]
        [MaxLenght(30)]
        public string Customer { get; set; }
        [Required]
```

```
        [MaxLenght(30)]
        public string Contact_person { get; set; }
        public DateTime Creation_date { get; set; }
        public DateTime Deadline { get; set; }
        [Required]
        [MaxLenght(10000)]
        public string Required_skills { get; set; }
    }
```

Б.3 Текст класу Customer

```
public class Customer
    {
        public int Id { get; set; }
        [Required]
        [MaxLenght(30)]
        public string Name { get; set; }
        [Required]
        [MaxLenght(30)]
        public string Status { get; set; }
        [Required]
        [MaxLenght(30)]
        public string Vacancies { get; set; }
        [Required]
        [MaxLenght(30)]
        public string Responsible_persons { get; set; }
        [Required]
        [MaxLenght(30)]
        public string Contacts { get; set; }
    }
```

Б.4 Текст класу Users

```
public class Users
    {
        public int Id { get; set; }
        [Required]
        [MaxLenght(30)]
        public string Name { get; set; }
        [Required]
        [MaxLenght(30)]
        public string Role { get; set; }
        [Required]
        [MaxLenght(30)]
```

```
        public string Region { get; set; }
        public DateTime Registration_date { get; set; }
    }
```

Б.5 Текст класу Candidate

```
public class Candidate
    {
        public int Id { get; set; }
        [Required]
        [MaxLenght(30)]
        public string Name { get; set; }
        [Required]
        [MaxLenght(30)]
        public string Position { get; set; }
        [Required]
        [MaxLenght(30)]
        public string Age { get; set; }
        [Required]
        [MaxLenght(30)]
        public string Country { get; set; }
        [Required]
        [MaxLenght(30)]
        public string Work_experience { get; set; }
        [Required]
        [MaxLenght(30)]
        public string Desired_salary { get; set; }
        [Required]
        [MaxLenght(30)]
        public string Source_candidate { get; set; }
    }
```

Б.6 Текст файлу AppDbContext.cs

```
public class AppDbContext:DbContext
    {
         public IDbSet<Vacancy> Vacancy { get; set; }
         public IDbSet<Candidate> Candidates { get; set; }
         public IDbSet<Customer> Customers { get; set; }
         public IDbSet<Users> Users { get; set; }
         public IDbSet<VacancyRequest> VacancyRequests { get; set; }
         public DataContext(DbContextOptions<DataContext> options)
             : base(options)
         {
```

```
public class DataBaseInitializer:DropCreateDatebaseAlways<AppDataDbContext>
protected override void Seed(AppDataDbContext context)
{
        var vacancy1 = new Vacancy
        { Name = “IOS developer”,
        Location = “Одеса”,
        Customer = “SoftSystem”,
        Salary = “за результатами співбесіди”,
        Open_date = new DateTime(year: 2022, month:08, day:20),
        Deadline = new DateTime(year: 2022, month:09, day:13),
        Responsible_recruiter = “Kolpakova Mariia”,
        Prioirty = “Високий”,
        Status = “Високий”,
        };

        var vacancy2 = new Vacancy
        { Name = “PHP developer”,
        Location = “Одеса”,
        Customer = “SoftSystem”,
        Salary = “за результатами співбесіди”,
        Open_date = new DateTime(year: 2022, month:06, day:20),
        Deadline = new DateTime(year: 2022, month:09, day:14),
        Responsible_recruiter = “Fedorova Mariia”,
        Prioirty = “Середній”,
        Status = “Високий”,
        };

        var vacancy3 = new Vacancy
        { Name = “Ruby developer”,
        Location = “Львів”,
        Customer = “SoftSystem”,
        Salary = “за результатами співбесіди”,
        Open_date = new DateTime(year: 2022, month:08, day:20),
        Deadline = new DateTime(year: 2022, month:09, day:23),
        Responsible_recruiter = “Kolpakova Viktoriia”,
        Prioirty = “Середній”,
        Status = “Низький”,
        };

        var vacancy4 = new Vacancy
        { Name = “Team Lead”,
        Location = “Тернопіль”,
        Customer = “SoftSystem”,
        Salary = “за результатами співбесіди”,
        Open_date = new DateTime(year: 2022, month:08, day:20),
        Deadline = new DateTime(year: 2022, month:09, day:26),
        Responsible_recruiter = “Kolpakova Mariia”,
        Prioirty = “Середній”,
```

```
Status = “Низький”,
};

var vacancy5 = new Vacancy
{ Name = “JS developer”,
Location = “Чернівці”,
Customer = “SoftSystem”,
Salary = “за результатами співбесіди”,
Open_date = new DateTime(year: 2022, month:08, day:20),
Deadline = new DateTime(year: 2022, month:09, day:27),
Responsible_recruiter = “Kolpakova Mariia”,
Prioirty = “Середній”,
Status = “Низький”,
};

var vacancy6 = new Vacancy
{ Name = “React developer”,
Location = “Вінниця”,
Customer = “SoftSystem”,
Salary = “за результатами співбесіди”,
Open_date = new DateTime(year: 2022, month:08, day:20),
Deadline = new DateTime(year: 2022, month:09, day:06),
Responsible_recruiter = “Kolpakova Mariia”,
Prioirty = “Середній”,
Status = “Низький”,
};

var vacancy7 = new Vacancy
{ Name = “WordPress developer”,
Location = “Київ”,
Customer = “SoftSystem”,
Salary = “за результатами співбесіди”,
Open_date = new DateTime(year: 2022, month:08, day:20),
Deadline = new DateTime(year: 2022, month:09, day:19),
Responsible_recruiter = “Kolpakova Mariia”,
Prioirty = “Середній”,
Status = “Низький”,
};

var vacancy8 = new Vacancy
{ Name = “Joomla developer”,
Location = “Одеса”,
Customer = “SoftSystem”,
Salary = “за результатами співбесіди”,
Open_date = new DateTime(year: 2022, month:08, day:20),
Deadline = new DateTime(year: 2022, month:09, day:22),
Responsible_recruiter = “Kolpakova Mariia”,
Prioirty = “Середній”,
```

```
Status = “Низький”,
};

var vacancy9 = new Vacancy
{ Name = “Shopify developer”,
Location = “Одеса”,
Customer = “SoftSystem”,
Salary = “за результатами співбесіди”,
Open_date = new DateTime(year: 2022, month:08, day:20),
Deadline = new DateTime(year: 2022, month:09, day:04),
Responsible_recruiter = “Kolpakova Mariia”,
Prioirty = “Середній”,
Status = “Низький”,
};

var vacancy10 = new Vacancy
{ Name = “IOS developer”,
Location = “Одеса”,
Customer = “SoftSystem”,
Salary = “за результатами співбесіди”,
Open_date = new DateTime(year: 2022, month:08, day:20),
Deadline = new DateTime(year: 2022, month:09, day:20),
Responsible_recruiter = “Kolpakova Mariia”,
Prioirty = “Середній”,
Status = “Низький”,
};

context.Vacancy.Add(vacancy1);
context.Vacancy.Add(vacancy2);
context.Vacancy.Add(vacancy3);
context.Vacancy.Add(vacancy4);
context.Vacancy.Add(vacancy5);
context.Vacancy.Add(vacancy6);
context.Vacancy.Add(vacancy7);
context.Vacancy.Add(vacancy8);
context.Vacancy.Add(vacancy9);
context.Vacancy.Add(vacancy10);

var users1 = new Users
{ Name = “Колпакова Вікторія Валеріївна”,
Role = “Рекрутер”,
Region = “Одеса”,
Registration_date = new DateTime(year: 2022, month:08, day:06),
};

var users2 = new Users
{ Name = “Колпакова Марія Олександрівна”,
Role = “Кандидат”,
```

```
Region = “Одеса”,
Registration_date = new DateTime(year: 2022, month:08, day:07),
};

var users3 = new Users
{ Name = “Миронюк Анна Олександрівна”,
Role = “Кандидат”,
Region = “Львів”,
Registration_date = new DateTime(year: 2022, month:08, day:09),
};

var users4 = new Users
{ Name = “Федорова Анна Олександрівна”,
Role = “Кандидат”,
Region = “Одеса”,
Registration_date = new DateTime(year: 2022, month:08, day:16),
};

var users5 = new Users
{ Name = “Леоненко Сергій Олександрович”,
Role = “Рекрутер”,
Region = “Тернопіль”,
Registration_date = new DateTime(year: 2022, month:08, day:10),
};

var users6 = new Users
{ Name = “Крамар Анна Володимирівна”,
Role = “Рекрутер”,
Region = “Одеса”,
Registration_date = new DateTime(year: 2022, month:08, day:03),
};

var users7 = new Users
{ Name = “Шевчук Сергій Володимирович”,
Role = “Кандидат”,
Region = “Львів”,
Registration_date = new DateTime(year: 2022, month:08, day:08),
};

var users8 = new Users
{ Name = “Антоненко Сергій Володимирович”,
Role = “Рекрутер”,
Region = “Одеса”,
Registration_date = new DateTime(year: 2022, month:08, day:18),
};

var users9 = new Users
{ Name = “Мироненко Сергій Володимирович”,
```

```
Role = “Кандидат”,
Region = “Львів”,
Registration_date = new DateTime(year: 2022, month:08, day:24),
};

var users10 = new Users
{ Name = “Колпаков Сергій Олександрович”,
Role = “Рекрутер”,
Region = “Львів”,
Registration_date = new DateTime(year: 2022, month:08, day:27),
};

var users11 = new Users
{ Name = “Стельмах Юлія Володимирівна”,
Role = “Кандидат”,
Region = “Львів”,
Registration_date = new DateTime(year: 2022, month:08, day:28),
};

context.Users.Add(users1);
context.Users.Add(users2);
context.Users.Add(users3);
context.Users.Add(users4);
context.Users.Add(users5);
context.Users.Add(users6);
context.Users.Add(users7);
context.Users.Add(users8);
context.Users.Add(users9);
context.Users.Add(users10);
context.Users.Add(users11);

var request1 = new VacancyRequest
{ Vacancy_name = “Joomla developer”,
Location = “Одеса”,
Customer = “SoftSystem”,
Contact_person = “Крамар Анна Володимирівна”,
Creation_date = new DateTime(year: 2022, month:08, day:20),
Deadline = new DateTime(year: 2022, month:09, day:31),
};

var request2 = new VacancyRequest
{ Vacancy_name = “DevOps”,
Location = “Львів”,
Customer = “SoftSystem”,
Contact_person = “Антоненко Юрій Володимирович”,
Creation_date = new DateTime(year: 2022, month:08, day:20),
Deadline = new DateTime(year: 2022, month:09, day:22),
};
```

```
var request3 = new VacancyRequest
{ Vacancy_name = “Java developer”,
Location = “Одеса”,
Customer = “SoftSystem”,
Contact_person = “Шевчук Сергій Володимирович”,
Creation_date = new DateTime(year: 2022, month:08, day:20),
Deadline = new DateTime(year: 2022, month:10, day:20),
};

var request4 = new VacancyRequest
{ Vacancy_name = “UI developer”,
Location = “Чернівці”,
Customer = “SoftSystem”,
Contact_person = “Стельмах Юлія Володимирівна”,
Creation_date = new DateTime(year: 2022, month:08, day:20),
Deadline = new DateTime(year: 2022, month:09, day:14),
};

var request5 = new VacancyRequest
{ Vacancy_name = “IT Project Manager”,
Location = “Суми”,
Customer = “SoftSystem”,
Contact_person = “Мироненко Сергій Володимирович”,
Creation_date = new DateTime(year: 2022, month:08, day:20),
Deadline = new DateTime(year: 2022, month:10, day:20),
};

var request6 = new VacancyRequest
{ Vacancy_name = “Архітектор глобальних рішень”,
Location = “Рівне”,
Customer = “SoftSystem”,
Contact_person = “Федорова Анна Олександрівна”,
Creation_date = new DateTime(year: 2022, month:08, day:20),
Deadline = new DateTime(year: 2022, month:09, day:20),
};

var request7 = new VacancyRequest
{ Vacancy_name = “Рекрутер”,
Location = “Львів”,
Customer = “SoftSystem”,
Contact_person = “Шевчук Олексій Володимирович”,
Creation_date = new DateTime(year: 2022, month:08, day:20),
Deadline = new DateTime(year: 2022, month:09, day:23),
};

var request8 = new VacancyRequest
{ Vacancy_name = “Тестувальник”,
Location = “Одеса”,
```

```
Customer = “SoftSystem”,
Contact_person = “Яковлев Валерій Валерійович”,
Creation_date = new DateTime(year: 2022, month:08, day:20),
Deadline = new DateTime(year: 2022, month:10, day:20),
};

var request9 = new VacancyRequest
{ Vacancy_name = “Проєктувальник баз даних”,
Location = “Львів”,
Customer = “SoftSystem”,
Contact_person = “Гончаренко Маргарита Валеріївна”,
Creation_date = new DateTime(year: 2022, month:08, day:20),
Deadline = new DateTime(year: 2022, month:09, day:22),
};

var request10 = new VacancyRequest
{ Vacancy_name = “Системний адміністратор”,
Location = “Одеса”,
Customer = “SoftSystem”,
Contact_person = “Леоненко Маргарита Іванівна”,
Creation_date = new DateTime(year: 2022, month:08, day:20),
Deadline = new DateTime(year: 2022, month:10, day:24),
};

context.VacancyRequest.Add(request1);
context.VacancyRequest.Add(request2);
context.VacancyRequest.Add(request3);
context.VacancyRequest.Add(request4);
context.VacancyRequest.Add(request6);
context.VacancyRequest.Add(request7);
context.VacancyRequest.Add(request8);
context.VacancyRequest.Add(request9);
context.VacancyRequest.Add(request10);

var candidate1 = new Candidate
{ Name = “Інженер-програміст”,
Position = “Middle”,
Age = “24”,
Country = “Україна”,
Work_experience = “2 роки”,
Desired_salary = “за результатами співбесіди”,
Source_candidate = “dou.ua”,
};

var candidate2 = new Candidate
{ Name = “Старший інженер-програміст”,
Position = “Senior”,
Age = “29”,
```

```
            Country = “Україна”,
            Work_experience = “5 років”,
            Desired_salary = “за результатами співбесіди”,
            Source_candidate = “dou.ua”,
            };

            var candidate3 = new Candidate
            { Name = “Молодший інженер-програміст”,
            Position = “Початківець”,
            Age = “21”,
            Country = “Україна”,
            Work_experience = “1 рік”,
            Desired_salary = “за результатами співбесіди”,
            Source_candidate = “dou.ua”,
            };

            var candidate4 = new Candidate
            { Name = “Архітектор рішень”,
            Position = “Senior”,
            Age = “34”,
            Country = “Україна”,
            Work_experience = “8 років”,
            Desired_salary = “за результатами співбесіди”,
            Source_candidate = “dou.ua”,
            };

            var candidate5 = new Candidate
            { Name = “Бізнес-аналітик”,
            Position = “Middle”,
            Age = “25”,
            Country = “Україна”,
            Work_experience = “3 роки”,
            Desired_salary = “за результатами співбесіди”,
            Source_candidate = “dou.ua”,
            };

            context.Candidate.Add(candidate1);
            context.Candidate.Add(candidate2);
            context.Candidate.Add(candidate3);
            context.Candidate.Add(candidate4);
            context.Candidate.Add(candidate5);

            context.SaveChanges();
        }
    }
}
```

ДОДАТОК В

Слайди презентації

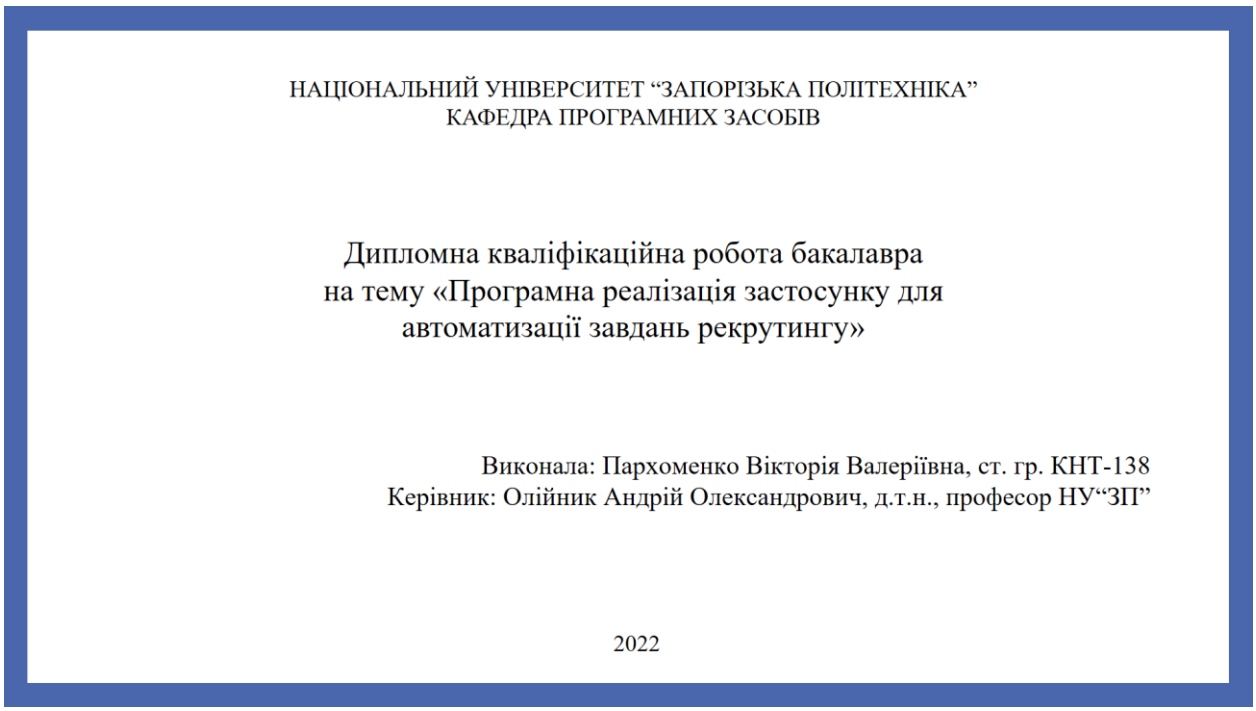

Рисунок В.1 – Титульний слайд

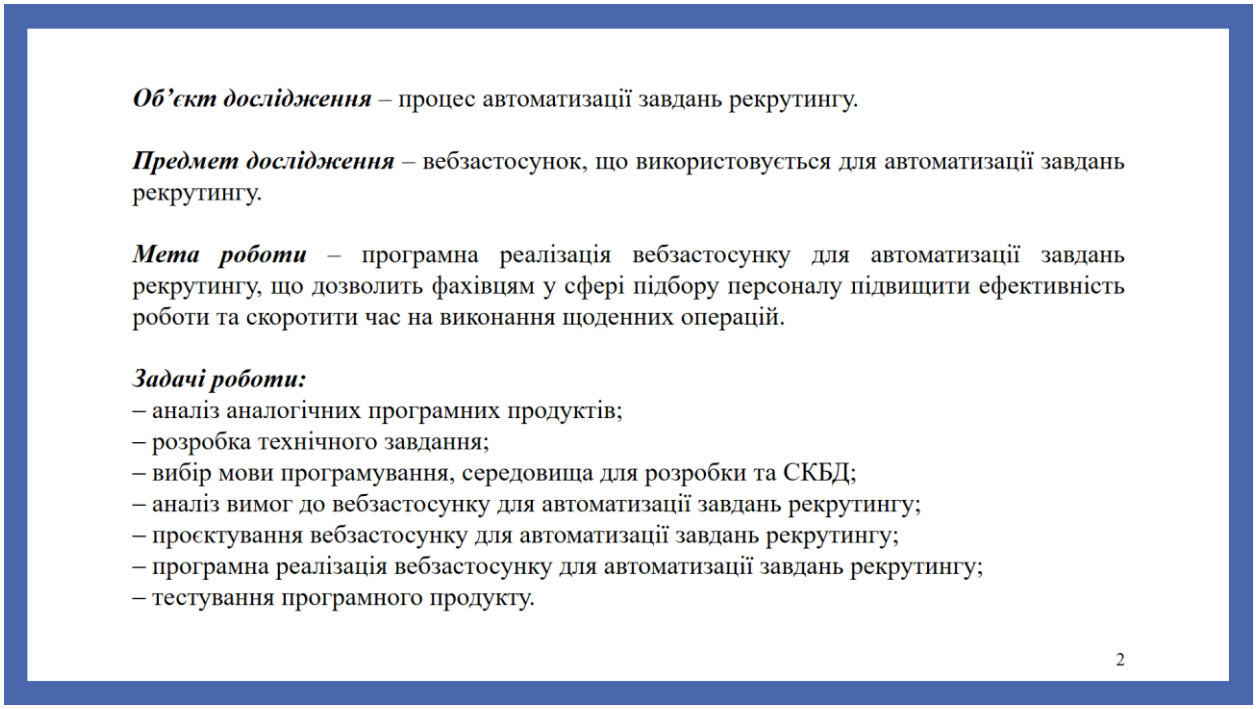

Рисунок В.2 – Об’єкт, предмет, мета роботи і задачі роботи

Постановка завдання роботи

Вебзастосунок для автоматизації завдань рекрутингу повинен надавати наступні функціональні можливості в залежності від типу користувача:
– аутентифікація облікового запису;
– перегляд активних вакансій;
– додавання, редагування та видалення вакансій;
– перегляд, створення та заповнення заявок на вакансію;
– узгодження або відхилення заявки на вакансію;
– перегляд бази замовників;
– додавання та видалення замовників;
– перегляд бази кандидатів;
– додавання, редагування та видалення кандидатів;
– перегляд бази користувачів;
– додавання, редагування та видалення користувачів.

3

Рисунок В.3 – Постановка завдання роботи

Порівняння сучасних рішень вирішення завдання

Програмний продукт / Критерій порівняння	«CleverStaff»	«Persia»	«Hurma»
Інтеграція з поштою	+	+	+
Інтеграція з календарем	+	+	+
Інтеграція з сайтами пошуку роботи	+	+	+
Розширення для браузерів	+	+	-
Статистика користувачів	+	+	+
Воронка рекрутингу	+	-	+
Конструктор звітів	+	-	-
Багатомовність	+	+	+
Кастомні поля	+	-	-
Публічна сторінка компанії	+	+	+
Платність	+	+	+

4

Рисунок В.4 – Порівняння сучасних рішень вирішення завдання

Порівняння мов програмування

Критерій порівняння \ Мова програмування	C#	PHP	Python	Ruby
Багатогранність розробки	++	-	+	+
Статистична типізація	+	-	-	-
Легкість освоєння	+	-	+	+
Багатопарадигмальна мова	++	+	+	+
Компільований тип мови	+	-	-	-
Мова високого рівня	+	+	+	+
Об'єктно-орієнтована мова	+	+	+	+
Швидкість виконання	+	+	-	-
Мультипотоковість	+	-	+	+
Модифікатор доступу до методу	+	+	-	+
Асинхронність	+	-	+	-

5

Рисунок В.5 – Порівняння мов програмування

Порівняння серверних фреймворків

Критерій порівняння \ Фреймворк	ASP.NET Core	Laravel
Мультиплатформеність	+	+
Будь-яка мова програмування	+	-
Велика кількість документації	+	+
Високий рівень безпеки	+	-
Придатність до великих розмірів рішень	+	-
Архітектурний шаблон MVC	+	+
Висока продуктивність роботи	+	-
Легкість освоєння	+	-
Компільований тип коду	+	-
Модульний підхід розробки	+	+

6

Рисунок В.6 – Порівняння серверних фреймворків

Порівняння середовищ для розробки

Середовище розробки / Критерій порівняння	Microsoft Visual Studio	JetBrains Rider
Мультиплатформеність	+	+
Підсвічування синтаксису	+	+
Підтримка GitLab	+	+
Аналіз коду	+	+
Менеджер пакетів NuGet	+	+
Провідник рішень	++	+
Технологія автодоповнення	++	+
Споживає небагато пам'яті	+	-
Швидкість відкриття рішення	++	+
Кодова навігація проєкту	++	+
Підтримка вбудованих баз даних	+	+
Модульне тестування	+	+

7

Рисунок В.7 – Порівняння середовищ для розробки

Порівняння СКБД

СКБД / Критерій порівняння	Microsoft SQL Server	PostgreSQL
Процедурні розширення SQL	Transact-SQL	PL/pgSQL
Безпека даних	+	-
Легкість освоєння	+	-
Велика кількість документації	+	-
Велика кількість підтримуваних типів даних	+	+
Зручний інтерфейс	+	-
Реляційна модель даних	+	+
Масштабованість	+	-
Клієнт-серверна архітектура	+	+
Необмежений розмір бази даних	+	+
Висока продуктивність роботи	+	-

8

Рисунок В.8 – Порівняння СКБД

Рисунок В.9 – Архітектура вебзастосунку

Рисунок В.10 – Діаграма прецедентів

Рисунок В.11 – Модулі реалізованого вебзастосунку

Рисунок В.12 – Схема структури бази даних

Рисунок В.13 – Алгоритм функціонування вебзастосунку

Рисунок В.14 – Реалізація інтерфейсу адміністратора

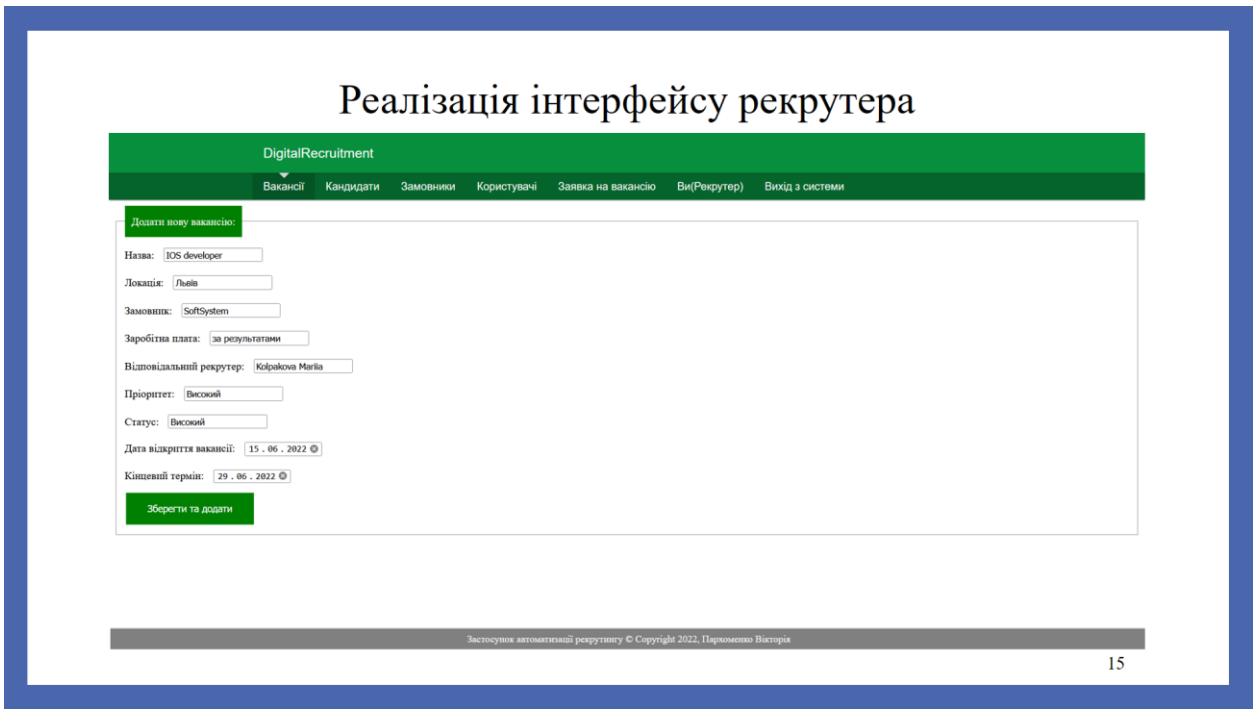

Рисунок В.15 – Реалізація інтерфейсу рекрутера

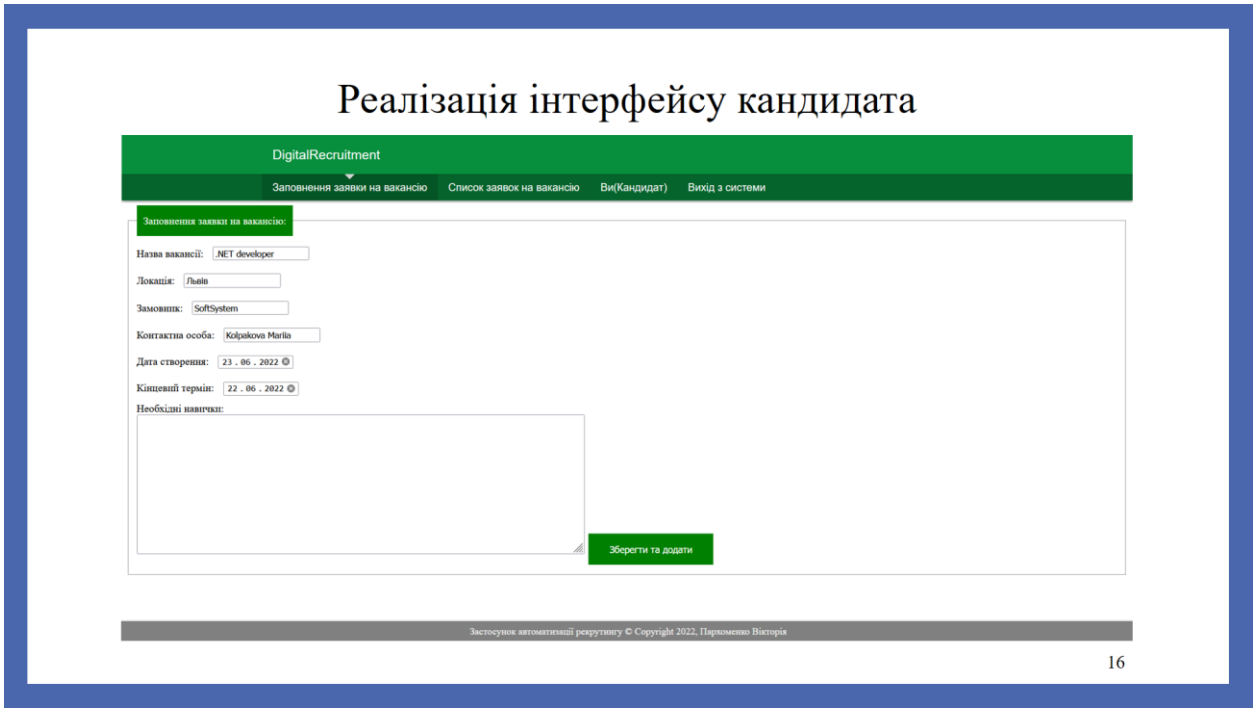

Рисунок В.16 – Реалізація інтерфейсу кандидата

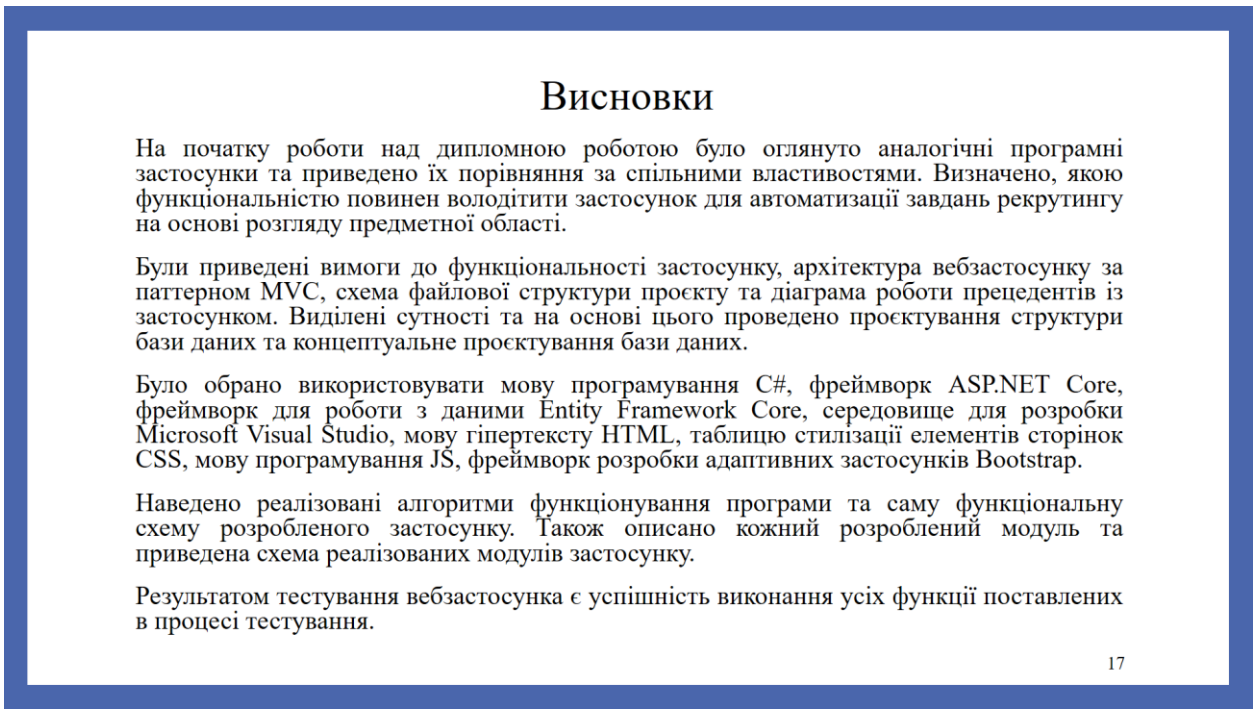

Рисунок В.17 – Висновки