Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет комп'ютерних наук
(повна назва)

Кафедра програмної інженерії
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

Програмна система для автоматизації онлайн-рекрутингу. DevOps, Full-Stack.
(тема)

Виконав:
студент 4 курсу, групи ПЗПІ-20-5

Клименюк Е.С.
(прізвище, ініціали)

Спеціальність 121 – Інженерія програмного забезпечення
(код і повна назва спеціальності)
Тип програми освітньо-професійна
Освітня програма Програмна інженерія
(повна назва освітньої програми)

Керівник доц. кафедри ПІ Побіженко І.О.
(посада, прізвище, ініціали)

Допускається до захисту
Зав. кафедри _____ З.В.Дудар
(підпис) (прізвище, ініціали)

2024 р.

Харківський національний університет радіоелектроніки

Факультет комп'ютерних наук
Кафедра програмної інженерії
Рівень вищої освіти перший (бакалаврський)
Спеціальність 121 – Інженерія програмного забезпечення
Тип програми Освітньо-професійна
Освітня програма Програмна Інженерія
(шифр і назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«___» _____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студента Клименюка Едуарда Сергійовича
(прізвище, ім'я, по батькові)

1. Тема роботи Програмна система для автоматизації онлайн-рекрутингу. DevOps, Full-Stack.

Затверджена наказом по університету від
20.05.2024 № 471 Ст

2. Термін подання студентом роботи до екзаменаційної комісії 19.06.2024

3. Вихідні дані до роботи у програмній системі передбачити: Створення серверної частини на платформі .NET 6, публічний API, розробка клієнтської частини на фреймворку AngularTs, налаштування розгортання додатку в хмарі, автоматизація рекрутингового процесу, захист системи. Використовувати JetBrains Rider, JetBrains WebStorm, JetBrains DataGrip, Docker Desktop.

4. Перелік питань, що потрібно опрацювати в роботі

Вступ, аналіз предметної галузі, формування вимог до програмної системи, архітектура та проектування програмного забезпечення, опис прийнятих програмних рішень, тестування розробленого програмного забезпечення, висновки, додатки.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Аналіз предметної галузі	8.04.2024	*виконано*
2	Створення специфікації ПЗ	22.05.2024	*виконано*
3	Проектування ПЗ	24.05.2024	*виконано*
4	Розробка ПЗ	28.05.2024	*виконано*
5	Тестування ПЗ	30.05.2024	*виконано*
6	Оформлення пояснювальної записки	05.06.2024	*виконано*
7	Підготовка презентації та доповіді	06.06.2024	*виконано*
8	Попередній захист	09.06.2024	*виконано*
9	Нормоконтроль, рецензування	06.06.2024	*виконано*
10	Здача роботи у електронний архів	14.06.2024	*виконано*
11	Допуск до захисту у зав. кафедри	19.06.2024	*виконано*

Дата видачі завдання 4 квітня 2024р.

Студент (ка)_____ Клименюк Е.С.
(підпис)

Керівник роботи _____ доц. кафедри ПІ Побіженко І.О.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ / ABSTRACT

Пояснювальна записка до кваліфікаційної роботи бакалавра: 72 стор., 14 рис., 7 джерел.

АВТОМАТИЗОВАНА РОЗРОБКА, AZURE, AZURE DEVOPS, DEVOPS, DATAGRIP, RIDER, WEBSTORM, YAML.

Об'єкт розробки - програмна система для автоматизації онлайн-рекрутингу, що включає в себе розгортання додатку на платформі Azure та налаштування автоматичного деплою. Також розробка серверної частини та клієнтської частин програми.

Мета розробки – спрямована на поліпшення ефективності та організації процесу найму персоналу, автоматизація процесів та покращення використання ресурсів за допомогою сучасних технологій та платформ.

Метод рішення – середовище розробки Rider, WebStorm, Azure DevOps, мова програмування YAML, хмарна платформа Azure.

У результаті розробки було створено програмну систему для автоматизації онлайн-рекрутингу, яка включає в себе розгортання додатку в хмарному середовищі Azure та налаштування автоматичного деплою.

AZURE DEVOPS, AUTOMATED DEVELOPMENT, DEVOPS, YAML, AZURE, RIDER, WEBSTORM

The object of development - a software system for online recruiting automation, which includes deploying an application on the Azure cloud and setting up automatic deployment.

The goal of the development is to improve the efficiency and organization of recruiting process, automate processes, optimize and improve the use of resources through modern technologies and platforms.

Method of solution - Rider development environment, WebStorm, Azure DevOps, .NET Core, C# programming language, Azure cloud services.

As a result of the development, a software system was created to automate online recruiting, which includes deploying the application in the Azure cloud and configuring automated deployment process.

Я, Клименюк Едуард Сергійович, студент гр. ПЗПІ-20-5, здобувач вищої освіти на першому (бакалаврському) рівні кафедри «Програмна інженерія», заявляю: моя кваліфікаційна робота на тему «Програмна система для автоматизації онлайн-рекрутингу. DevOps», що буде представлена в екзаменаційну комісію для публічного захисту, виконана самостійно, в ній не містяться елементи плагіату і вона може бути опублікована в електронному архіві відкритого доступу ElAr KhNURE. Усі запозичення з друкованих та електронних джерел мають відповідні посилання.

Я ознайомлений(а) з діючим положенням «Про протидію академічному плагіату в ХНУРЕ», згідно з яким виявлення плагіату є підставою для відмови до допуску кваліфікаційної роботи до захисту та застосування дисциплінарних заходів.

ЗМІСТ

ВСТУП

Програмна система для автоматизації онлайн-рекрутингу є потужним інструментом для компаній, які займаються наймом персоналу. Проте, подібні програми потребують ефективного управління, організації та координації, що вимагає відповідної програмної системи для ефективного виконання цих завдань.

Основною метою цього проекту є розробка програмної системи автоматизації онлайн-рекрутингу. Система буде розгорнута на Azure де буде налаштований автоматичний деплой. Така система повинна поліпшити ефективність та організацію процесу найму персоналу, надаючи зручні інструменти для планування, координації, відстеження та звітності щодо вакансій та кандидатів.

Додаток, повинен відповідати сучасним стандартам, які відповідають додаткам з автоматизації процесу рекрутингу. З точки зору програмної частини, система повинна бути незалежною, оскільки це буде частиною роботи з розробки програмного забезпечення для автоматизації процесу рекрутингу.

Сфера в якій буде використовуватися система – рекрутинг та підбір персоналу. Розроблені, в рамках комплексної роботи, системи можуть зацікавити рекрутенгових спеціалістів, великі компанії та людей в інших сферах діяльності. Підсумковий результат повинен бути доступний для використання на велику аудиторію користувачів і поширюватися безкоштовно або за доступною для користувачів регіональною ціною, що сприятиме розвитку системи.

Розроблене програмне забезпечення повинне мати можливість автоматичного розгортання на сервері Azure за допомогою Azure DevOps.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1 Аналіз предметної галузі

Автоматизація онлайн-рекрутингу є важливим елементом оптимізації рекрутингових процесів і забезпечує успішний підбір та управління талантами. Розробка програмного забезпечення для цієї галузі передбачає використання DevOps підходу для автоматизації різних етапів, включаючи розробку, тестування, розгортання та моніторинг. Одним із ключових аспектів цього процесу є розгортання серверу на хмарній платформі Azure з використанням автоматичного деплою.

Деплой програми на платформі Azure включає в себе вибір потрібного сервісу для розгортання. Залежно від ситуації можна обрати Azure Virtual Machine, Azure Container Service чи Azure App Service для веб-додатків [2].

Автоматичне розгортання забезпечується за допомогою різних інструментів, таких як Azure Pipelines. Це включає автоматичну зборку додатку, проведення тестування, упаковку та розгортання на продукційні сервери, а також автоматизацію розгортання додатків.

Ці стратегії та інструменти DevOps сприяють зростанню ефективності, скороченню часу випуску продукту та підвищенню стійкості системи. Це критично для системи, що автоматизує рекрутинг, оскільки вона повинна бути надійною та готовою до розширення, щоб задовольняти потреби значної аудиторії користувачів.

Процес безперервної інтеграції передбачає регулярне об'єднання різних версій коду в один основний потік розробки. Ця стратегія сприяє виявленню та виправленню помилок на ранніх етапах, що дозволяє ефективно зменшити час, потрібний для їх вирішення.

Безперервна доставка сприяє оперативному впровадженню змін, забезпечуючи гнучкість і ефективність. Код завжди готовий до впровадження в основну систему, що дозволяє командам оперативно реагувати на зміни в бізнес-процесах або ринку.

Це також зменшує ризики, оскільки впроваджувані зміни є контрольованими і мінімізованими.

Безперервний моніторинг допомагає забезпечувати високу якість та доступність системи, надаючи командам оперативну інформацію про стан застосунку та його середовища. Це включає такі метрики, як продуктивність, навантаження на систему,

час відгуку та помилки. Ця інформація може бути використана для виявлення та усунення проблем, прогнозування й запобігання майбутніх неполадок, а також для планування потреб у масштабуванні.

Ці підходи разом створюють ефективну та гнучку систему розробки та впровадження програмного забезпечення, яка може відповідати вимогам управління системи автоматизації рекрутингу.

1.2 Аналіз існуючих рішень

У сучасному світі, де технології швидко розвиваються, важливо не лише створювати якісні продукти, а й адекватно оцінювати конкурентні можливості. Дослідження конкурентів дозволяє краще зрозуміти ринок, виявити сильні та слабкі сторони інших учасників і визначити, що робить наш продукт унікальним, якими можливостями він володіє, щоб виділитися серед конкурентів. У цьому аналізі ми порівняємо двох ключових конкурентів у сфері програмних систем для автоматизації рекрутингу.

Першим конкурентом є LinkedIn Talent Solutions . LinkedIn Talent Solutions надає платформу для пошуку, залучення та найму кандидатів у різних галузях. Їхні інструменти включають автоматизоване розміщення вакансій, аналіз профілів кандидатів і інтеграцію з іншими системами управління персоналом.

Другим конкурентом є Greenhouse. Greenhouse - це інша платформа для рекрутингу, яка надає інструменти для керування всім процесом найму, від розміщення вакансій до оцінки кандидатів. Вони пропонують функціонал для автоматизації етапів відбору, планування співбесід і ведення обліку даних про кандидатів.

У порівнянні з цими двома конкурентами, виявляються значні різниці в функціональності. Однак, якщо говорити про технологічний аспект, системи для автоматизації рекрутингу можуть підвищити свою конкурентоспроможність, використовуючи платформу Azure для розгортання та впровадження сучасних методик DevOps.

1.3 Виявлення проблем

Після розгляду основних конкурентів для нашої системи, є сенс виділити їх плюси та мінуси почнемо з LinkedIn Talent Solutions:

а) плюси LinkedIn Talent Solutions:

1) широкі можливості пошуку – LinkedIn Talent Solutions надає доступ до великої бази професійних профілів, що дозволяє здійснювати широкий і ефективний пошук кандидатів за різними критеріями, що може підвищити швидкість знаходження відповідних співробітників;
2) інтеграція з професійними мережами – LinkedIn Talent Solutions інтегрується з професійною соціальною мережею LinkedIn, що дозволяє отримати більш повну інформацію про кандидатів та їх професійний досвід;
3) аналітика та звіти – платформа надає можливості для аналізу ефективності рекрутингових кампаній та створення звітів, що дозволяє підвищити ефективність процесів найму.

б) мінуси LinkedIn Talent Solutions:

1) високі витрати – використання LinkedIn Talent Solutions може бути дорогим для деяких компаній, особливо для невеликих бізнесів або стартапів;
2) залежність від якості профілів – пошук кандидатів на LinkedIn Talent Solutions ґрунтується на якості їхніх профілів у мережі LinkedIn, що може обмежувати доступні опції в залежності від рівня професійної активності користувачів;
3) конкуренція – Змагання за увагу кандидатів на платформі LinkedIn може бути великим, особливо в популярних галузях або для вакансій з високим попитом.

Тепер можна перейти до переваг та недоліків наступного Greenhouse:

а) плюси Greenhouse:

1) налагоджена система керування кандидатами – Greenhouse надає широкі можливості для управління процесом найму, включаючи резюме кандидатів, організацію співбесід, внутрішню комунікацію та аналіз

результатів;

2) працює на основі даних – платформа надає засоби для аналізу даних та статистики найму, що дозволяє підвищити ефективність процесу та прийняття рішень;

б) мінуси Greenhouse:

1) складність використання – деякі користувачі відзначають складність інтерфейсу та процесу налаштування, що може вимагати додаткового часу та зусиль для оволодіння платформою;

2) залежність від інтеграції – повна функціональність Greenhouse може бути досягнута лише через інтеграцію з іншими інструментами та системами, що може створювати проблеми з сумісністю та налаштуванням.

Тепер, коли ми розглянули основних конкурентів що схожі за нашим проектом, ми можемо перейти до наступного етапу - розробки постановки задачі.

1.4 Постановка задачі

Ефективне управління процесом онлайн-рекрутингу є ключовим елементом успішної реалізації наймальних програм. Розробка програмного забезпечення, яке допомагатиме організаторам у цьому процесі, потребує використання DevOps підходу для автоматизації процесів розробки, тестування, розгортання та моніторингу. Важливим етапом є розгортання серверу на платформі Azure з використанням автоматичного деплою.

Система управління автоматизованим онлайн-рекрутингом має на меті забезпечити ефективне керування різними аспектами цього процесу, включаючи реєстрацію кандидатів, розподіл завдань, ведення звітності та надання необхідної підтримки. Для досягнення цієї мети, важливо використовувати підхід, що сприяє спільній роботі розробників, тестувальників та адміністраторів, а також автоматизує процеси розробки та розгортання.

Метою проекту є створення програмної системи, яка забезпечить ефективне управління процесом онлайн-рекрутингу з використанням підходу DevOps[1]. Головні цілі проекту включають:

– автоматизоване розгортання на платформі Azure: сервер буде автоматично розміщений у середовищі Azure з метою забезпечення надійності,

продуктивності та безпеки. Це включає встановлення необхідного програмного забезпечення, налаштування мережевих параметрів та впровадження політик безпеки;

- моніторинг та логування: це важливі складові для забезпечення стабільності та надійності системи. Для цього потрібно створити рішення, яке дозволить відстежувати роботу системи та збирати події в централізованому журналі;
- неперервну інтеграцію та доставку (CI/CD): використання автоматизованих процесів для компіляції, тестування та розгортання програмного коду на сервер. Це дозволяє швидко внести нові зміни та оновлення безпосередньо у середовище;
- тестування: виконання тестів для впевненості в правильній роботі системи, розгортання та працездатності.

Налаштування розгортання сервера на платформі Azure разом із використанням автоматичного деплою дозволить ефективно керувати програмним середовищем. Це сприятиме швидкій ітерації, стабільності та простоті впровадження нових версій програмної системи.

2 ФОРМУВАННЯ ВИМОГ ДО ПРОГРАМНОЇ СИСТЕМИ

2.1 Функціональні вимоги

Необхідно належним чином розробити складову частину роботи, оскільки її функціональність буде використовуватися в усіх інших етапах проекту.

Для створення високоякісного програмного продукту, зокрема веб-додатку, необхідно чітко визначити функціональні вимоги. Серед функціональних вимог для деплою в рамках проекту включаються:

- система управління конфігураціями;
- система автоматичного деплою;
- система моніторингу та логування;
- система управління версіями.

Давайте детальніше розглянемо ці системи, починаючи з системи управління конфігураціями. Вона включає такі функції:

- зберігання конфігураційних файлів: ця функція забезпечує збереження всіх конфігураційних файлів проекту в одному централізованому місці;
- версіонування конфігурацій: система повинна дозволяти версіонувати конфігураційні файли, щоб зберігати історію змін та забезпечувати можливість відновлення попередніх версій;
- автоматизація розгортання конфігурацій: ця функція дозволяє автоматизувати процес розгортання конфігурацій на середовищі веб-додатку.

Система автоматичного деплою передбачає реалізацію таких функцій:

- конфігурація середовища деплою: система повинна мати можливість налаштовувати середовище, на яке буде проводитися автоматичне розгортання програмного забезпечення;
- автоматизоване розгортання програмних компонентів: функція дозволяє автоматично розгортати нові версії програмного забезпечення на визначене середовище без необхідності втручання оператора;
- резервне копіювання: система може забезпечувати автоматичне створення резервних копій перед розгортанням нової версії програмного забезпечення для забезпечення можливості відновлення в разі необхідності.

Система моніторингу та логування повинна мати наступні функції:

- збір даних про стан системи: система повинна здійснювати постійний моніторинг різних параметрів системи, таких як використання ресурсів (пам'ять, CPU), доступність сервісів, швидкість відповіді тощо;
- збереження логів подій: система повинна забезпечувати збереження логів подій, що відбуваються в системі, таких як запити користувачів, помилки, відповіді сервера тощо;
- сповіщення про проблеми: система повинна надавати можливість налаштування сповіщень про виникнення проблем або критичних ситуацій в системі для оперативного реагування.

Через те, що ці системи будуть інтегровані одна з одною, вони повинні виступати єдиним цілим, сприяючи легкому управлінню та високій якості продукту.

Далі перейдемо до функціональних вимог під час розробки клієнтської та серверної частини додатку.

Завантаження резюме під час подання на вакансію має кілька значних переваг як для кандидатів, так і для роботодавців. По-перше, це забезпечує швидкість та зручність процесу подання заявки. Кандидати можуть легко завантажити вже підготовлене резюме, що значно скорочує час, необхідний для заповнення онлайн-форм та інших документів. Це дозволяє їм швидше реагувати на нові вакансії та збільшує шанси на швидку відповідь від роботодавця.

Для роботодавців завантажене резюме забезпечує стандартизацію та порівнянність заявок. Всі кандидати надають інформацію в уніфікованому форматі, що полегшує процес порівняння кваліфікацій та досвіду. Це також спрощує процес попереднього відбору, дозволяючи рекрутерам швидко визначати відповідність кандидатів вимогам вакансії.

Таким чином, завантаження резюме під час подання на вакансію забезпечує швидкість та зручність процесу, стандартизацію та порівнянність заявок, надання повної інформації про кваліфікації кандидата, зручність зберігання та обробки даних, покращує взаємодію між кандидатами та роботодавцями, а також дозволяє кандидатам краще презентувати себе. Ці переваги роблять цей процес важливою складовою ефективного рекрутингу.

Програмна система повинна надсилати електронні листи, якщо користувач не читає повідомлення в чаті. Це дозволить користувачам завжди бути в курсі, чи не пропустили вони важливе повідомлення.

Важливою частиною функціоналу є порівняння заявки користувача на вакансію. Це дозволить користувачу швидко визначити, чи відповідає він вимогам для даної позиції.

2.2 Нефункціональні вимоги

Система повинна бути гнучкою та здатною швидко реагувати на зміни у програмному коді та швидко впроваджувати ці зміни для кінцевого користувача. Нефункціональні вимоги до програмного забезпечення у рамках розгортання можуть включати:

- продуктивність та масштабованість, щоб забезпечити ефективну роботу системи навіть під час великого обсягу даних або навантаження;
- безпека та доступність для захисту від несанкціонованого доступу та забезпечення неперервної доступності системи для користувачів;
- автоматизація процесів розгортання та управління для підвищення ефективності та зменшення можливості помилок.

Давайте розглянемо ці вимоги детальніше, а саме почнемо з продуктивності та масштабованості:

- обробка великих обсягів даних та користувачів: платформа повинна бути здатною ефективно обробляти великі обсяги даних та велику кількість користувачів без зниження продуктивності. Це включає швидку обробку запитів, оптимізацію ресурсів та забезпечення коректної відповіді на запити при будь-яких навантаженнях;
- легка масштабованість: система повинна мати можливість легко масштабуватися для підтримки зростання та змін потреб користувачів. Це означає, що платформа повинна бути гнучкою та можливою до розширення, додавання ресурсів і компонентів, щоб відповідати зростаючому обсягу даних та навантаженню користувачів без значних затримок або перебоїв у роботі.

Безпека та доступність:

- захист даних користувачів та системи: система повинна забезпечувати високий рівень захисту даних користувачів та системи в цілому. Це включає в себе застосування шифрування, механізми контролю доступу, виявлення та запобігання злому, а також захист від вразливостей;
- система авторизації та аутентифікації: повинна бути реалізована система авторизації та аутентифікації для забезпечення безпеки доступу до системи та її компонентів. Це означає, що користувачі повинні проходити процедуру аутентифікації перед отриманням доступу, а їх права доступу повинні бути належним чином налаштовані та контрольовані;
- безперебійне розгортання та оновлення: розгортання та оновлення системи повинні відбуватися безперебійно, з мінімальним впливом на доступність системи. Це означає, що процеси поновлення та розгортання повинні бути ретельно плановані та виконані таким чином, щоб уникнути відмов та збоїв у роботі системи під час цих процесів.

Автоматизація та управління включають в себе:

- автоматичні процеси: система має мати автоматизовані процеси для розгортання, моніторингу, тестування та виправлення помилок, що спрощує керування та зменшує час, необхідний для управління системою;
- інтуїтивно зрозумілі інструменти: система повинна мати зрозумілі інструменти для управління та моніторингу, які дозволяють керувати системою та виявляти та виправляти проблеми легко.

Нефункціональні вимоги для програмної системи онлайн-рекрутингу, що розгортається на платформі Azure, мають велике значення, оскільки вони визначають ефективне використання системи як розробниками, так і кінцевими користувачами. Ці вимоги сприяють поліпшенню враження від системи, роблять її більш доступною та зручною у використанні, а також забезпечують її стабільність і надійність. Крім того, вони гарантують відповідність системи вимогам якості продукту, що мають ключове значення.

2.2 Вимоги до середовища розгортання

Для створення та розгортання нашої системи ми вирішили скористатися обчислювальною платформою Microsoft Azure. Azure пропонує широкий спектр

сервісів та можливостей, що дозволяють розробляти, впроваджувати та масштабувати додатки в хмарному середовищі.

Для написання програмного коду ми використали інтегроване середовище розробки WebStorm, створене компанією JetBrains. WebStorm [5] має вбудовану підтримку мови програмування YAML, що дозволяє створювати Azure Pipelines.

Однією з ключових переваг WebStorm є його вбудований допоміжний функціонал, швидкодія та вбудований функціонал відладки помилок.

Використовуючи Azure DevOps як сервіс для керування репозиторіє, ми отримуємо не лише засоби контролю версій, але й спромогу розробити процеси автоматизації за допомогою Azure Pipelines. Завдяки цьому можна позбутися таких операцій як тестування та розгортання, що є важливою складовою DevOps.

З цього виходить, що вибір Azure як сервісу для отримання рахувальних можливостей, WebStorm для написання програмного коду та Azure DevOps як систему контролю версій та пайплайнів, допоміг створити якісне та ефективне середовище для розробки системи автоматизації онлайн-рекрутингу.

3 АРХІТЕКТУРА І ПРОЕКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Діаграми є потужним інструментом при розробці програмного забезпечення з кількох причин. Вони допомагають візуалізувати складні системи та процеси, роблячи їх зрозумілішими. Це особливо важливо для великих проектів, де безліч компонентів та взаємозв’язків між ними. Діаграми також полегшують комунікацію між членами команди, оскільки різні фахівці, такі як розробники, тестувальники та менеджери проектів, можуть швидше зрозуміти загальну картину та деталі проекту. Крім того, діаграми служать як документація, допомагаючи зберегти знання про систему, що є корисним при обслуговуванні та оновленні ПЗ, особливо коли нові члени команди приєднуються до проекту.

Діаграми дозволяють проводити аналіз системи на ранніх етапах розробки, виявляти потенційні проблеми та знаходити рішення до того, як код буде написаний, що економить час та ресурси. Вони допомагають розбити складні системи на більш зрозумілі частини, що полегшує їхнє проектування та розробку. Існують різні типи діаграм для різних стадій розробки ПЗ, такі як UML діаграми для моделювання об’єктно-орієнтованих систем, діаграми потоків даних для аналізу бізнес-процесів та діаграми архітектури для планування структури системи. Використання діаграм сприяє більш структурованому, організованому та ефективному процесу розробки ПЗ, що веде до більш якісного кінцевого продукту.

3.1 UML проєктування програмного забезпечення

Під час розробки програмного забезпечення для автоматизації онлайн-рекрутингу, було створено глобальну UML Use Case діаграму. Детальний вигляд якої представлено на рисунку 3.1 Особлива увага приділяється основним діям та акторам, які є учасниками теми: "Розгортання серверу на платформі Azure та налаштування автоматичного деплою". На зображенні нижче відображено загальну Use Case діаграму системи, що включає в себе акторів і дії, пов'язані з темою. Серед акторів виступають Developer, Azure DevOps та Azure. Давайте далі детальніше розглянемо дії кожного з акторів, починаючи з одного з головних діючих обличь цієї діаграми – Developer. Developer - це актор, який є фізичною особою та працює над проектом. Він розробляє додаток, проводить його тестування, робить правки та використовує Azure

DevOps, як місце управління версіями програмного коду та взаємодії з іншими членами команди.

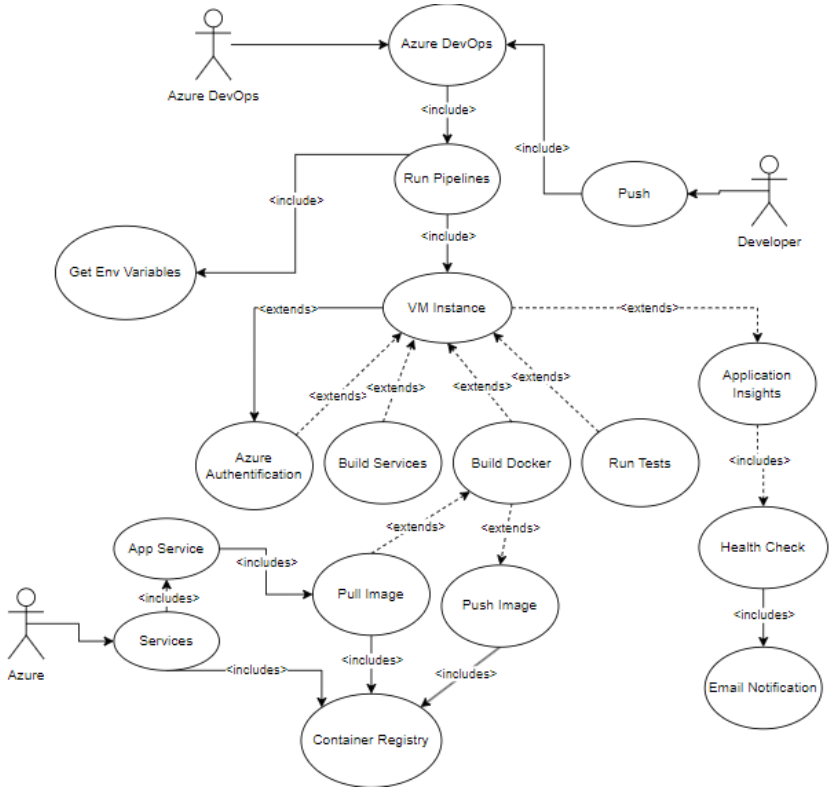


Рисунок 3.1 – Use Case діаграма програмного застосунку (рисунок виконаний самостійно)

Azure DevOps – цей актор виступає інструмент для управління версіями, який дозволяє розробникам спільно працювати над проектом, відстежувати зміни, відновлювати попередні версії та робити оновлення коду незалежно один від одного та одночасно.

Azure – це актор у UML діаграмі, що представляє хмарну інфрастуктуру від Microsoft. Ця інфраструктура використовується для підтримки, розгортання, оновлення, управління та налагодження програмного забезпечення.

Після ідентифікації акторів, розглянемо їх основні дії та розберемо їх докладніше. Дії розподілимо на Developer (Розробник), Azure DevOps та Azure. Розглянемо дії, які мають вагомий внесок під час розробки додатку:

- Push Code (відправити код) – команда git push завантажує коміти з локальної гілки до відповідної гілки у віддаленому репозиторії. Вона оновлює віддалений репозиторій з локальними змінами, забезпечуючи синхронізацію

роботи з іншими розробниками. Команда зазвичай використовується після виконання команди git commit, щоб зроблені локальні зміни стали доступними на віддаленому сервері.

- Дія "Run Pipelines" (Виконання конвеєрів) у Azure DevOps відіграє ключову роль у процесі неперервної інтеграції та доставки (CI/CD). Конвеєри активуються автоматично після того, як розробник надсилає зміни в репозитарій у Azure DevOps. Це може бути зроблено через команду git push, що відправляє оновлення коду з локальної машини на сервер. Конвеєри містять ряд завдань, які визначені для обробки коду, його збірки, тестування, створення контейнерів та інших операцій, які необхідні для підготовки програмного забезпечення до розгортання. Цей етап є фундаментальним для підтримки неперервності та якості процесу розробки програмного забезпечення в автоматизованому та контрольованому середовищі;
- Дія "Get Env Variables" (Отримання змінних середовища) є важливим компонентом для забезпечення коректної конфігурації та безпечного використання секретних даних під час процесів автоматизації. Змінні середовища використовуються для зберігання конфігураційних параметрів, які можуть варіюватися між середовищами (наприклад, розробка, тестування, продакшн) або зберігання секретних даних, таких як API ключі, паролі, тощо. Під час виконання конвеєрів, система автоматично витягує значення змінних середовища і використовує їх для конфігурації завдань, що виконуються. Наприклад, шляхи до баз даних, необхідні параметри для з’єднань, чи конфігураційні файли, що мають бути використані під час розгортання. Під час виконання конвеєрів, система автоматично витягує значення змінних середовища і використовує їх для конфігурації завдань, що виконуються. Наприклад, шляхи до баз даних, необхідні параметри для з’єднань, чи конфігураційні файли, що мають бути використані під час розгортання. Цей етап є ключовим для налаштування та безпеки різних процесів у конвеєрах і забезпечує ефективну інтеграцію та взаємодію різних компонентів системи у різних середовищах;
- Application Insights – ця дія виконується в контексті "Run Pipelines", та є інструментом для моніторингу додатків у Azure, який збирає детальну

інформацію про продуктивність, використання ресурсів, винятки, помилки і транзакції користувачів. На базі даних, зібраних Application Insights, можуть бути налаштовані сповіщення, які активуються при певних умовах, наприклад, коли показники продуктивності виходять за межі заданих порогів;

- Health Check (перевірка стану) – ця дія виконується в контексті "Run Pipelines", виконує регулярні перевірки стану різних компонентів додатка, щоб гарантувати їх правильну роботу та доступність. Це може включати перевірку доступності веб-сервісів, баз даних, інтеграцій з зовнішніми сервісами та інше;
- Email Notification (повідомлення у вигляді електронного листа) – ця дія виконується в разі генерації виключення під час виконання "Health Check". При виявленні проблем під час виконання Health Check або коли моніторинг в Application Insights фіксує певні події, може бути автоматично надіслано Email Notification відповідальним особам або команді підтримки. Ці сповіщення можуть містити детальну інформацію про проблему, що спрощує процес діагностики та швидкого вирішення інцидентів;
- Run Tests (виконати тести) – є критичним для забезпечення якості коду і правильності його функціонування перед розгортанням на продакшн;
- Build Services (побудувати сервіси) – є фундаментальним для процесу розробки та розгортання програмного забезпечення. Цей етап включає компіляцію коду, підготовку бінарних файлів, та інші завдання, що забезпечують готовність програми до розгортання та тестування. Цей крок забезпечує, що усі необхідні компоненти будуть доступні разом з програмою під час її розгортання. Всі необхідні ресурси, такі як бібліотеки, зображення, файли конфігурації, та інші залежності, збираються та пакуються разом із скомпільованим кодом;
- Azure Authentication (аутентифікація в Azure) – є ключовим аспектом безпеки, який забезпечує захищений доступ до ресурсів і сервісів в Azure. Цей процес включає встановлення ідентичності користувачів, сервісів або додатків перед наданням доступу до ресурсів Azure. В основі аутентифікації в Azure лежить Azure Active Directory (Azure AD), який є обліковою системою для всіх користувачів, груп, і сервісів;

- Build Docker (збірка Docker) – стосується створення Docker образів із застосунками чи сервісами. Це дозволяє стандартизувати середовища виконання застосунків та спростити процес розгортання. Образи можуть бути побудовані автоматично в рамках CI/CD конвеєру в Azure DevOps, забезпечуючи безперервність процесів розробки та доставки;
- Push Image (відправка образу) – дія котра відділена від інших та стосується відправлення зібраних Docker образів до реєстру контейнерів, такого як Azure Container Registry. Цей крок є ключовим для забезпечення доступності образів для подальшого розгортання в різних середовищах;
- Закінчивши опис дій, котрі виконує актор - Azure DevOps, перейдемо до актора Azure:
- Azure Services (сервіси Azure) – цей етап стосується управління та взаємодії з розгорнутими сервісами в хмарному середовищі. Розгортання сервісів включає налаштування та активацію програмного забезпечення у середовищі Azure. Це може включати веб-додатки, фонові служби, бази даних, інтеграційні сервіси тощо;
- Azure App Service (сервіс додатку) – дозволяє легко розгортати веб-додатки за допомогою FTP, Git, Mercurial, або як Docker контейнери. Також можна імпортувати код безпосередньо з таких репозитаріїв, як GitHub, BitBucket, або Azure DevOps. Сервіс дозволяє розробникам швидко публікувати та оновлювати додатки без складностей управління інфраструктурою;
- Azure Container Registry (реєстр контейнерів) – є службою управління Docker контейнерними образами, яка дозволяє зберігати та управляти приватними Docker образами для всіх типів контейнерних розгортань. Використання ACR інтегрується з Azure Kubernetes Service (AKS), Azure App Service та іншими сервісами Azure, забезпечуючи безпечний та надійний шлях для зберігання образів, які використовуються в продуктивному середовищі.
- Pull Image (завантаження образу) – відноситься до процесу завантаження Docker образів з реєстру до локальної системи або до середовища розгортання. Цей процес є ключовим для розгортання та управління контейнеризованими застосунками. Ось детальний опис цієї операції.

Результат виконання цього етапу використовує App Service, для того щоб розгорнути додаток;

- Дії, які ми обговорили, представляють ключові етапи та аспекти під час розгортання додатку за допомогою Azure та Azure DevOps. Ці кроки ілюструють необхідні дії, які треба виконати різними акторами під час розробки, тестування та розгортання.

3.2 Вибір архітектури та хмарних рішень

На сьогоднішній день існує багато хмарних сервісів, що відрізняються за наданою функціональністю та складністю в використанні. Основними гравцями на світовому ринку є Amazon Web Services (AWS) [3] і Microsoft Azure.

AWS надає широкий спектр послуг для обчислень, зберігання даних, баз даних, аналітики, штучного інтелекту, Інтернету речей (IoT), безпеки та розгортання додатків. Вона дозволяє компаніям та розробникам швидко та ефективно створювати, розгортати та масштабувати свої додатки. Крім того, він має глобальну мережу серверних центрів, що забезпечує швидкий доступ та низьку затримку для користувачів з усього світу.

Microsoft Azure пропонує широкий вибір інструментів для розгортання, керування та розширення веб-застосунків та послуг. Azure відомий своєю інтеграцією з іншими продуктами Microsoft, що робить його зручним для користувачів, які вже використовують ці продукти. Він також має глобальне присутність серверів, що забезпечує швидкий доступ та надійність послуг для користувачів у будь-якій точці світу.

Нами було обрано Microsoft Azure для подальшої розробки нашої системи для оптимізації процесу онлайн-рекрутингу, так як наша система ідеально підходить під цього провайдера хмарних послуг. Azure надасть усі сервіси, за допомогою яких можна буде впровадити автоматичне розгортання, спираючись на стратегії DevOps.

Microsoft Azure надає широкий спектр інструментів для автоматизації процесів розгортання та впровадження програмного забезпечення. Використання цих інструментів дозволило нам прискорити час випуску нових версій програм, підвищити якість продукту та збільшити продуктивність нашої команди. Тому ми вибрали Microsoft Azure для нашого проекту.

Використовуючи оптимальні стратегії архітектури, можна побудувати систему для автоматизації онлайн-рекрутингу та забезпечити ефективне управління нею наступним чином.

Головним аспектом у розробці додатку є його програмний код. Шляхом використання передових методик розробки, таких як контроль якості та систематичні оновлення системи, ми можемо забезпечити високу якість кінцевого продукту, зменшити ймовірність помилок та спростити подальше управління кодом.

Після розробки коду він пройде через процес "будівництва", що включає компіляцію, тестування та пакування для майбутнього розгортання. Наша мета полягає в автоматизації цього процесу та забезпеченні його надійності, щоб забезпечити безперебійну роботу системи.

Перед випуском продукту в продакшн він повинен пройти ретельне тестування. Цей підхід включає проведення модульних та навантажувальних тестів з метою переконатися, що продукт працює правильно і зможе витримати прогнозоване навантаження.

Після успішного проходження всіх тестів розпочинається процес деплою додатку. Це можливо автоматизувати за допомогою різних практик, таких як неперервна інтеграція/неперервне розгортання.

Після впровадження продукту необхідно забезпечити його постійний нагляд та технічну підтримку. Це включає моніторинг продуктивності, виявлення та виправлення неполадок, а також регулярне оновлення та вдосконалення продукту. Ефективне впровадження процедур управління інцидентами та вирішення проблем є критичним для реагування на будь-які непередбачені ситуації.

Для забезпечення неперервного контролю над усіма аспектами системи, включаючи продуктивність, надійність та безпеку, важливим аспектом є моніторинг. Це дозволяє вчасно виявляти та вирішувати проблеми, а також виявляти можливості для покращення.

Останнім етапом у нашій архітектурі є постійне вдосконалення. Це означає, що ми уважно слідкуємо за результатами нашої роботи, шукаємо можливості для поліпшення та впроваджуємо їх у наші процеси. Цей підхід дозволяє нам удосконалювати систему автоматизації онлайн-рекрутингу, забезпечуючи більшу ефективність та надійність.

Цей процес передбачає постійне оновлення та адаптацію до новітніх технологій і найкращих практик, щоб підвищити продуктивність та якість нашої системи. Також ми постійно аналізуємо дії та відгуки користувачів, це дає змогу зрозуміти, як покращити їх користувацький досвід.

За допомогою цієї архітектури ми можемо створити систему для оптимізації, яка буде ефективною, надійною та легкою. Вона забезпечить безперебійну роботу та можливість масштабування відповідну до потреб системи.

3.3 Розгляд найцікавіших алгоритмів

Задля економії місця, через розміри рисунка 3.2, алгоритм було скорочено. Однак, у наступному розділі ви знайдете детальний опис цього алгоритму, який допоможе вам отримати більш повне розуміння процесу.

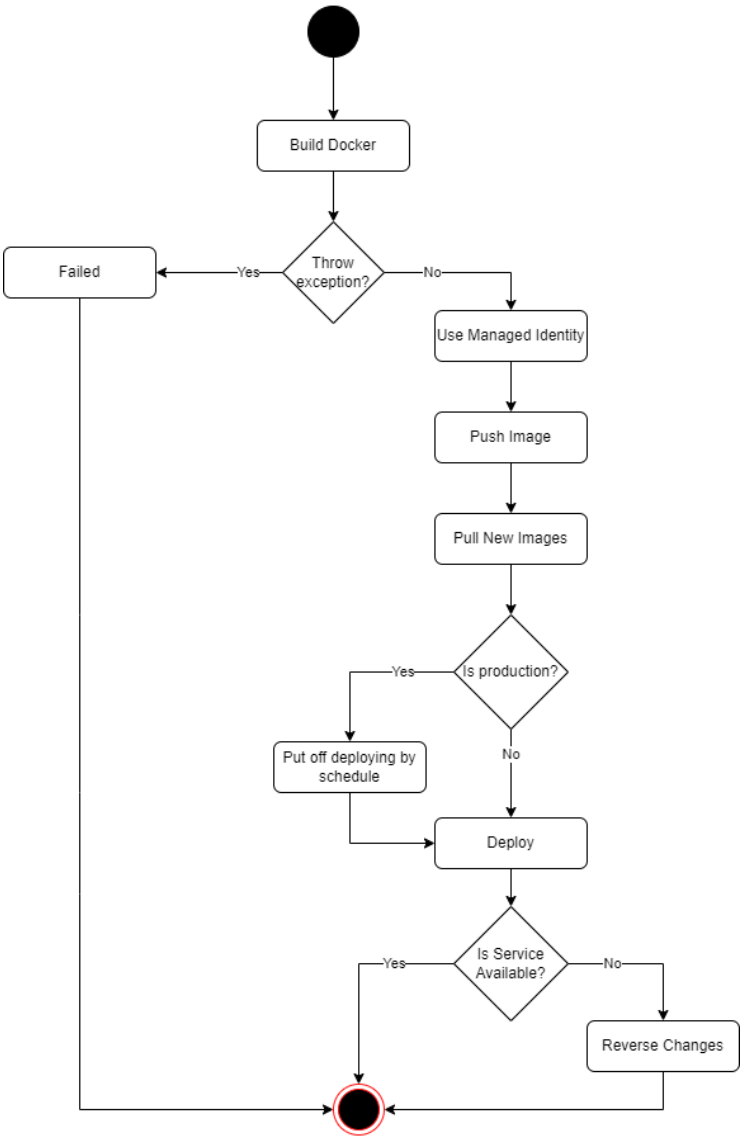


Рисунок 3.2 – Алгоритм деплою Azure (рисунок виконаний самостійно)

Першим кроком алгоритму створення Docker-образу. Це крок, де все починається, і він має ключове значення для успішного розгортання додатку Будівництво Docker включає створення Docker-образу з Dockerfile. Dockerfile — це текстовий файл, що містить всі команди, які користувач може викликати в командному рядку для збірки образу. Він визначає, яке програмне забезпечення та які налаштування будуть використовуватися в середовищі всередині контейнер.

Другий етап "Повернення виключення" (Throw Exception) є важливим моментом для контролю помилок під час процесу збірки образу Docker. Цей крок стосується рішення про те, як обробляти помилки, які можуть виникнути на попередньому етапі. Цей крок забезпечує відмовостійкість розгортання, допомагаючи запобігти розповсюдженню недоліків або помилково сконфігурованих образів у продакшн середовище. Він виступає важливим бар'єром безпеки, що забезпечує, що помилки виявляються та обробляються належним чином ще до впровадження змін.

Крок "Використання керованого ідентифікатора" (Use Managed Identity) у діаграмі алгоритму розгортання Docker відіграє важливу роль у процесі автоматизації та безпеки. Цей крок зосереджений на використанні керованих ідентифікаторів для авторизації та управління доступом до ресурсів. Цей крок забезпечує високий рівень безпеки та зручність управління доступом до ресурсів у хмарних середовищах, дозволяючи розробникам зосередитись на логіці додатку, не переймаючись з управлінням облікових даних та доступом.

Наступний етап у діаграмі алгоритму розгортання Docker, позначений як "Відправлення образу" (Push Image), є критичним кроком у процесі управління Docker-образами. Цей етап є важливим для забезпечення швидкості та ефективності розгортання додатків:

доступність: завантажені образи стають доступні для розгортання в будь-якому середовищі, де є доступ до репозиторію, що дозволяє швидко масштабувати додатки.

версіонування: тегування допомагає управляти версіями образів, що спрощує процес відкату до попередньої стабільної версії, якщо потрібно.

Цей крок дозволяє розробникам та системним адміністраторам мати постійний контроль над версіями та розгортанням додатків, забезпечуючи, що будь-яке середовище, від розробки до продакшну, може бути легко оновлене з останньою версією образу.

Наступний крок позначений як "Завантаження нових образів" (Pull New Images), є критичною частиною процесу розгортання. Цей крок забезпечує актуалізацію образів на серверах або в середовищах, де буде виконуватися розгортання. Завантаження останніх образів гарантує, що всі екземпляри додатку працюють із схожими налаштуваннями та кодом, що знижує ймовірність помилок, пов'язаних з різницею конфігурацій. Оновлення образів забезпечують впровадження останніх патчів безпеки та виправлень помилок, що підвищує загальний рівень безпеки системи. Цей крок дозволяє розробникам та системним адміністраторам підтримувати актуальність середовищ розробки, тестування та продакшену, забезпечуючи гладке та ефективне розгортання додатків.

Крок "Визначення середовища для деплою" в алгоритмі розгортання Docker відіграє ключову роль, оскільки він дозволяє оптимізувати використання ресурсів та забезпечує гнучкість управління різними середовищами — від розробки до продакшну. Цей крок дозволяє здійснити глибше тестування перед впровадженням в продакшн, що сприяє підвищенню якості додатку.

Якщо обране середовище є продакшн, то наступний важливий етап — "Перевірка доступності сервісу після розгортання" стає критичним для забезпечення належної роботи додатку. Цей крок підтверджує, що всі компоненти додатку функціонують належним чином і сервіс доступний для кінцевих користувачів. Якщо на цьому етапі виявляються будь-які проблеми, команда може швидко відреагувати та вирішити їх, що гарантує неперервність доступу до сервісу.

Алгоритм розгортання є добре структурованим і враховує важливі аспекти безпеки, стабільності та гнучкості в управлінні розгортанням програмного забезпечення. Він охоплює всі критичні кроки, від збірки образів до їх розгортання в відповідних середовищах, перевірки їх доступності та реагування на можливі помилки. Загалом, цей алгоритм розгортання покращує якість розгортання програмного забезпечення та ефективність управління інфраструктурою, дозволяючи організаціям бути більш гнучкими та реактивними на зміни вимог та середовища.

3.4 Створення основних Azure Services

Спочатку було створено інфраструктуру на платформі Azure для забезпечення автоматичного деплою.

Resource Group в рамках Azure є логічним контейнером, який використовується для управління ресурсами Azure. Вона дозволяє організовувати і управляти всіма ресурсами, пов'язаними з певним проектом або додатком, у межах одного логічного блоку. Кожен ресурс в Azure, такий як віртуальні машини, бази даних, мережі тощо, повинен належати до однієї ресурсної групи. Ресурсна група дозволяє управляти життєвим циклом усіх ресурсів як однієї сутності. Це означає, що ви можете розгортати, оновлювати та видаляти всі ресурси в групі одночасно, що значно спрощує управління. Використовуючи ресурсні групи, ви можете логічно організовувати ресурси за проектами, додатками або середовищами (наприклад, розробка, тестування, виробництво), що допомагає підтримувати порядок і структуру у ваших ресурсах.

У рамках проекту створено групу ресурсів під назвою "RecruitXpert", яка виступає контейнером для всіх сервісів Azure, що використовувалися під час розробки та впровадження системи, включаючи Azure App Service, Azure Container Registry, Azure Managed Identity та Azure SQL Server, Storage Account, Application Insights.

Створення групи ресурсів “RecruitXpert” дозволяє логічно поєднати всі засоби, що використовуються в системі, в одному місці, спрощуючи управління, моніторинг та забезпечення безпеки. Вона спрощує управління життєвим циклом ресурсів, дозволяючи розгортати, оновлювати та видаляти всі ресурси в групі одночасно. Крім того, це дозволяє швидко і зручно масштабувати систему, оскільки все необхідне розташоване в одному місці, як показано на рисунку 3.3.

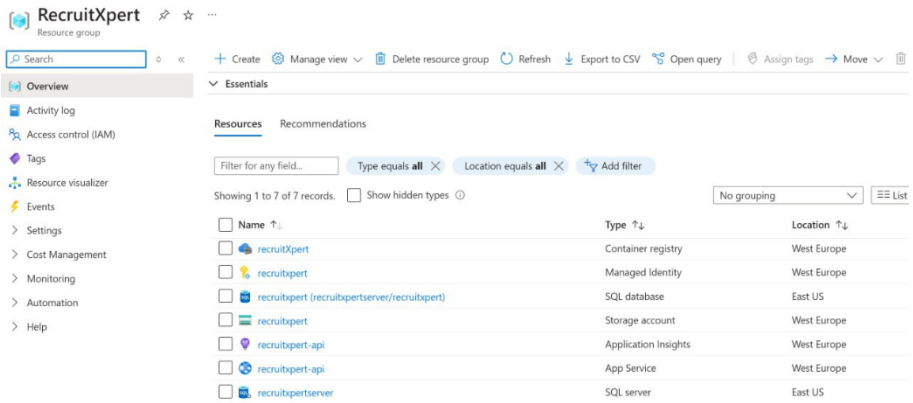


Рисунок 3.3 – RecruitXpert Resource Group (рисунок виконаний самостійно)

На рисунку 3.4 показаний Azure Managed Identity. Managed Identity спрощує взаємодію з Azure Container Registry, забезпечуючи безпечний та автоматичний доступ

до реєстру контейнерів. Зазвичай, для взаємодії з контейнерним реєстром потрібні облікові дані, такі як ім'я користувача та пароль. Проте, використання Azure Managed Identity усуває необхідність в ручному керуванні цими обліковими даними.

Azure Managed Identity дозволяє додаткам та службам безпечно аутентифікуватися з іншими службами Azure без необхідності керувати обліковими даними вручну. Вона автоматично надає керовані ідентичності, які можна використовувати для отримання доступу до ресурсів Azure, таких як Key Vault, Azure SQL Database або Storage, з мінімальними налаштуваннями. Azure Managed Identity допомагає усунути ризики, пов'язані з управлінням секретами та обліковими даними, забезпечуючи безпечний спосіб аутентифікації додатків і служб у хмарному середовищі.

Azure Managed Identity надає автоматичний токен доступу для пайплайну чи сервісу, що працює з контейнерним реєстром. Цей токен дозволяє автоматично автентифікувати пайплайн або сервіс у Azure Container Registry під час проведення операцій з контейнерами, такими як завантаження, оновлення або видалення. Такий підхід забезпечує безпеку, оскільки відпадає потреба у вбудовуванні облікових даних в конфігурацію, і спрощує управління, оскільки ви не маєте стежити за цими обліковими даними та їх оновленням.

Azure Managed Identity спрощує взаємодію з Azure Pipelines шляхом автоматизації процесу надання доступу до ресурсів Azure. Зазвичай, для взаємодії з Azure ресурсами з CI/CD пайплайнами потрібно використовувати облікові дані. Проте, з використанням Azure Managed Identity, ця задача стає набагато простішою та безпечнішою.

Azure Managed Identity надає пайплайнам автоматичні облікові дані доступу до ресурсів Azure. При цьому не потрібно жорстко вбудовувати облікові дані в конфігурації пайплайну. Замість цього, Azure Managed Identity надає пайплайну токен доступу, який автоматично автентифікує його в Azure під час виконання завдань деплою. Це забезпечує безпеку, оскільки не потрібно зберігати облікові дані в конфігурації пайплайну, і зростає ефективність, оскільки виключається необхідність керування та оновленням цих облікових даних.

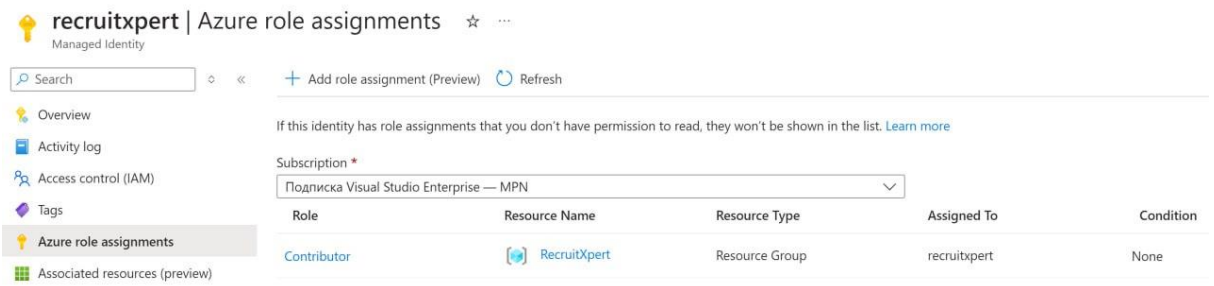


Рисунок 3.4 – RecruitXpert Identity (рисунок виконаний самостійно)

Azure Container Registry дозволяє зберігати та керувати контейнерними образами для всіх типів контейнерних розгортань. Ця служба забезпечує безпечне та масштабоване сховище для Docker-образів, підтримуючи інтеграцію з іншими службами Azure, такими як Azure Kubernetes Service (AKS) і Azure DevOps. Azure Container Registry допомагає автоматизувати робочі процеси збірки та розгортання, забезпечуючи при цьому контроль версій та управління доступом. Використання цієї служби спрощує розгортання контейнерів у хмарі, дозволяючи легко управляти та розповсюджувати контейнерні образи в межах організації.

Azure Container Registry ідеально вписується в екосистему інструментів, таких як Azure DevOps, Jenkins та Docker CLI, і має широкі можливості для взаємодії з Azure DevOps. Це робить процес зберігання та розповсюдження контейнерних образів для наших додатків дуже простим та зручним, незалежно від того, де ми знаходимося в процесі розробки.

Ми налаштували автоматичну збірку та оновлення контейнерних образів у Azure Container Registry безпосередньо з Azure DevOps під час кожного оновлення репозиторію. Це забезпечує постійну актуальність наших контейнерних образів і відображає останні зміни у коді.

Продакшн база даних вважається невід'ємною складовою будь-якої системи, оскільки вона містить всі життєво важливі дані, які обробляє система. Azure SQL Server пропонує набір функцій, що спрямовані на забезпечення стабільності, продуктивності та безпеки продакшн бази даних.

Azure SQL Server дозволяє створювати, управляти та масштабувати реляційні бази даних у хмарі. Це керована служба баз даних, яка забезпечує високу доступність, автоматичне резервне копіювання, безпеку даних та відновлення після збоїв. Azure SQL Server підтримує автоматичне масштабування ресурсів, що дозволяє легко

адаптувати продуктивність бази даних до потреб вашого додатку. Інтеграція з іншими службами Azure, такими як Azure Machine Learning і Power BI, дозволяє легко аналізувати дані та будувати бізнес-аналітику. Azure SQL Server спрощує управління базами даних, дозволяючи зосередитися на розробці додатків без необхідності витрачати час на налаштування та обслуговування інфраструктури баз даних.

Система побудована на двох Azure Web App (App Service). Перший виступає як API, на рисунку 3.5, а другий як frontend.

Azure Web App Service дозволяє розгортати, масштабувати та керувати веб-додатками в хмарі.

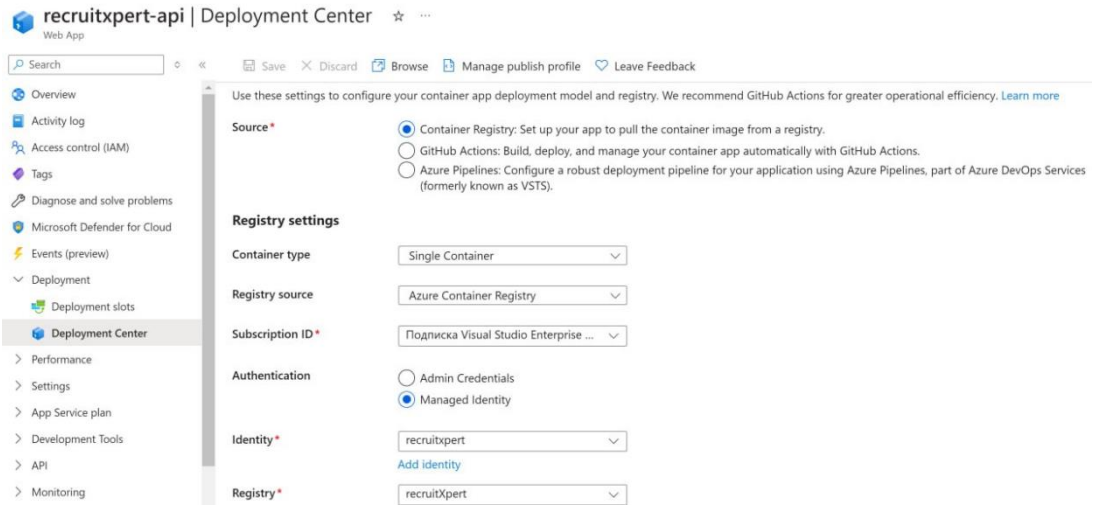


Рисунок 3.5 – RecruitXpert API (рисунок виконаний самостійно)

Ця керована платформа забезпечує високу доступність, автоматичне масштабування, безпеку та управління резервними копіями, що дозволяє розробникам зосередитися на написанні коду, не турбуючись про інфраструктуру. Azure Web App Service підтримує різні мови програмування, такі як .NET, Java, Python, PHP і Node.js, а також інтегрується з інструментами CI/CD для автоматизації розгортання. Використання цієї служби спрощує процес управління веб-додатками, забезпечуючи швидке і безпечне розгортання, а також легке масштабування відповідно до зростаючих потреб бізнесу.

Обидва сервіси використовують контейнерні образи з Azure Container Registry, що є центральним джерелом істини для коду та його залежностей. Це спрощує процес

синхронізації коду між розробкою, тестуванням та продакшн середовищами, а також полегшує розгортання оновлень та виправлень.

Також, завдяки Azure Web App, ми можемо автоматично масштабувати та керувати ресурсами на рівні платформи. Це сприяє забезпеченню стабільності та продуктивності обох сервісів, незалежно від навантаження.

Застосування Azure Web App [5] для впровадження двох сервісів (API та frontend) в системі для оптимізації онлайн-рекрутингу сприяє гнучкості, безпеці та простоті управління, що сприяє підтримці надійності та ефективності додатку.

При розгортанні "Програмної системи для оптимізації онлайн-рекрутингу" на платформі Azure, використання таких сервісів, як Azure Web App, Azure Container Registry, Azure Managed Identity, Azure SQL Server, Application Insights та Storage Account, дозволило створити надійну, масштабовану та безпечну інфраструктуру. Використання цих сервісів Azure спрощує процес розробки, розгортання та управління додатками, дозволяючи команді зосередитися на розробці функціоналу системи, а не на управлінні інфраструктурою. Це підтверджує ефективність Azure як платформи для розробки та розгортання сучасних веб-додатків та сервісів.

3.5 Розробка Azure Pipelines

Для автоматизації процесу розгортання, було використано Azure Pipelines. Azure Pipelines [6] дозволяє автоматизувати збірку, тестування та розгортання додатків за допомогою інтегрованого процесу безперервної інтеграції та доставки (CI/CD). Ця служба підтримує різні платформи та мови програмування, забезпечуючи гнучкість для різноманітних проектів. Azure Pipelines інтегрується з популярними системами контролю версій, такими як GitHub та Azure Repos, що дозволяє автоматично запускати збірку та тестування при кожному оновленні коду. Вона також підтримує паралельні збірки та розгортання в різних середовищах, включаючи хмару та локальні ресурси. Використання Azure Pipelines допомагає прискорити розробку, підвищити якість програмного забезпечення та забезпечити швидке та надійне розгортання додатків. Під час розробки функціоналу автоматичного деплою, Azure Pipelines були основним інструментом.

Azure Pipelines мають певний набір подій, які можна налаштувати наприклад на оновлення певної гілки Git репозиторію. При тригері цих івентів, Azure Pipelines

запускають визначені послідовні події. Їх можна використати для створення Azure Services\Resources, оновлення образів в Azure Container Registry та ін.

Давайте детальніше розглянемо пайплайн (див. рис. 3.6) котрий відповідає за оновлення образів нашого додатку, котрі зберігаються в Azure Container Registry.

Цей пайплайн Azure DevOps автоматизує процес зборки та публікації Docker-контейнерів для двох компонентів: API та клієнтського додатку.

Команда **pool**, є основою будь-якого Azure Pipeline та вказує на пул агентів, які будуть використовуватися для виконання завдань в пайплайні, в більшості випадків пулом агентів виступають операційні системи такі як Windows, Linux. Тут використовується стандартний пул агентів Azure Pipelines, який в свою чергу бере конфігурацію з основи пайплайну.

```
pool:
  name: Azure Pipelines
steps:
- task: Docker@2
  displayName: 'Build and Push API'
  inputs:
    containerRegistry: 'Container registry'
    repository: api
    Dockerfile: RecruitXpert/Recrutire.API/Dockerfile
    buildContext: RecruitXpert

- task: Docker@2
  displayName: 'Build and Push Client'
  inputs:
    containerRegistry: 'Container registry'
    repository: client
    Dockerfile: Recrutire.UI/Dockerfile
    buildContext: Recrutire.UI
```

Рисунок 3.6 RecruitXpert-Docker container-CI (рисунок виконаний самостійно)

Давайте розглянемо першу команду task, котра відповідає за оновлення образу серверної частини додатку.

Команда **task,** Використовується завдання Docker@2, яке надає можливість взаємодіяти з Docker з метою збірки, тегування та публікації образів.

Команда **displayName**, 'Build and Push API' - опис завдання, яке вказує на те, що це крок для зборки та публікації API-контейнера.

Команда inputs та її компоненти:

- containerRegistry: 'Container registry' - вказує на ім'я Docker-реєстру, куди будуть публікуватися зібрані образи;
- repository: 'api' - назва репозиторію в контейнерному реєстрі, куди буде публікуватися образ;
- Dockerfile: 'RecruitXpert/Recrutire.API/Dockerfile' - шлях до Dockerfile, який використовується для зборки образу API;
- buildContext: 'RecruitXpert' - контекст зборки, який визначає кореневу директорію для процесу зборки Docker-образу.

Перейдемо до другої команди task, котра відповідає за оновлення образу клієнтської частини додатку.

Команда **task,** Використовується завдання Docker@2, яке надає можливість взаємодіяти з Docker з метою збірки, тегування та публікації образів.

Команда **displayName**, 'Build and Push API' - опис завдання, яке вказує на те, що це крок для зборки та публікації API-контейнера.

Команда inputs та її компоненти:

- containerRegistry: 'Container registry' - вказує на ім'я Docker-реєстру, куди будуть публікуватися зібрані образи;
- repository: 'client' - назва репозиторію в контейнерному реєстрі, куди буде публікуватися образ;
- Dockerfile: 'Recrutire.UI/Dockerfile' - шлях до Dockerfile, який використовується для зборки образу API;
- buildContext: 'Recrutire.UI' - контекст зборки, який визначає кореневу директорію для процесу зборки Docker-образу.

Переваги даного пайплайну:

- автоматизація: цей пайплайн автоматизує весь процес, зменшуючи кількість рутинної роботи та ймовірність помилок.;
- повторюваність: процес зборки та публікації контейнерів є повторюваним і може бути легко відтворений у будь-який час;
- інтеграція: використання Azure DevOps забезпечує інтеграцію з іншими інструментами та службами Azure, що полегшує управління та розгортання додатків.

Використання Azure Pipelines дозволяє прискорити, полегшити, уніфікувати та автоматизувати деплой системи. Завдяки усьому вище згаданому, під час процесу розробки ми отримуємо багато переваг, таких як прискорення розробки, меншу кількість конфліктних ситуацій, стандартизацію та автоматизацію.

3.6 Загрузка файлів

В нашій системі для автоматизації онлайн рекрутингу була передбачена можливість прикріплення свого резюме під час подання на вакансію. Місце, де зберігаються, файли, реалізоване за допомого Azure Storage Account (Blob Storage). Для того, щоб була можливість завантажувати файли, було розроблено API Endpoint(див. рис. 3.7). Також були розширені необхідні моделі, для того щоб вони могли зберігати в собі посилання на цей файл, і встановлено новий nuget package – Microsoft Azure Storage.

```
[Route( template: "api/[controller]")]
[ApiController]
[Authorize(AuthenticationSchemes = JwtBearerDefaults.AuthenticationScheme)]
Eduard Klymeniuk
public class BlobController : ControllerBase
{
    private readonly IBlobService _blobService;

    Eduard Klymeniuk
    public BlobController(IBlobService blobService){...}

    [AllowAnonymous]
    [RequestSizeLimit( bytes: 157_286_400)]//its relevant to 150 MB
    [HttpPost( template: "blob-upload")]
    Eduard Klymeniuk
    public async Task<IActionResult> UploadBlob([Required][FromQuery] string path, [Required][FromForm] IFormFile file)
    {
        var result = await _blobService.UploadToBlobAsync(path, file.FileName, file.OpenReadStream());

        return string.IsNullOrEmpty(result.Url) ? BadRequest( error: "File was not uploaded") : Ok(result);
    }
}
```

Рисунок 3.7 BlobController (рисунок виконаний самостійно)

Для того щоб надати контролеру необхідний функціонал було розроблено BlobService(див. рис. 3.8). Він включає в себе оптимізацію обробки великих файлів.

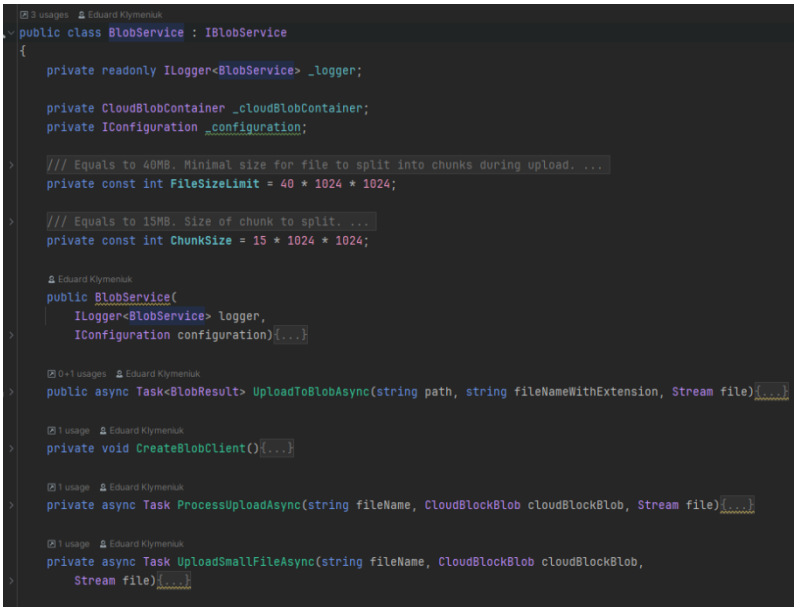


Рисунок 3.8 BlobService (рисунок виконаний самостійно)

Для того щоб оптимізувати час виконання запиту в разі того, якщо файл буде мати великі розміри, була використана бібліотека Microsoft.WindowsAzure.Storage. Вона надає необхідний набір функціоналу котрий довзоляє нам завантажувати файли по часткам. Алгоритм сервісу працює так, що він бере файл котрий прийшов до нього, потім перевіряє чи перевищує він ліміт розміру і якщо так, то починає розподіляти його на байтові порції (в подальшому чанки) та викликає метод цієї бібліотеки та загружає кожен чанк окремо в результаті чого процес відбувається більш оптимізованим для великих файлів.

3.7 Розсилка імейлів

В нашому додатку була реалізована система розсилки нагадувань у разі того якщо користувач не відповідає на повідомлення в чаті. Для цього була встановлена бібліотека HangFire котра допомагає реалізувати функціонал Background Jobs а також класс SubscriptionSender(див. рис. 3.9) котрий реєструється на рівні додатку та відпрацьовує кожні 12 годин.

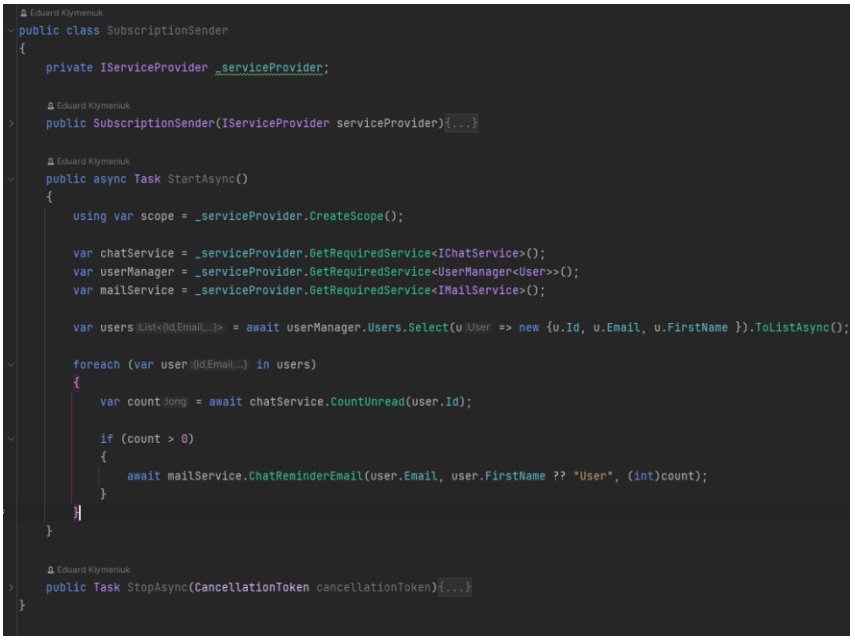


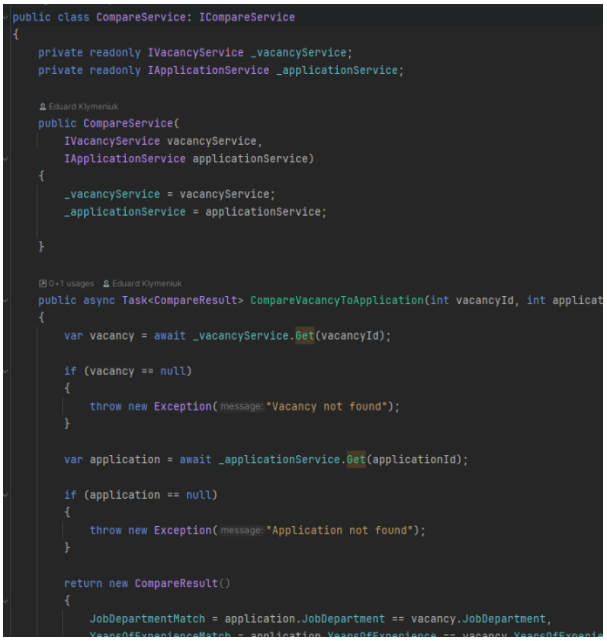
Рисунок 3.9 Subscription Sender (рисунок виконаний самостійно)

Завдяки розробці цього функціоналу, користувачі зможуть отримувати повідомлення в випадку того якщо вони не побачили повідомлення від іншого в чаті.

3.8 Порівняння вакансії до резюме

Функціонал, який дозволяє одразу побачити, чи підходить кандидат для вакансії на основі свого резюме, є надзвичайно корисним у нашому додатку. Це допомагає кандидату швидко зрозуміти, чи має він можливість успішно пройти співбесіду або ж влаштуватися на нову вакансію.

Для того щоб надати користувачу такий функціонал було розроблено окремий сервіс CompareService(див. рис. 3.10). Даний сервіс бере основні вагомі дані під час розгляду рекрутером резюме та порівнює їх з даними котрі є в резюме кандидата та повертає результат.



```
public class CompareService: ICompareService
{
    private readonly IVacancyService _vacancyService;
    private readonly IApplicationService _applicationService;

    Eduard Klymeniuk
    public CompareService(
        IVacancyService vacancyService,
        IApplicationService applicationService)
    {
        _vacancyService = vacancyService;
        _applicationService = applicationService;

    }

    0+1 usages  Eduard Klymeniuk
    public async Task<CompareResult> CompareVacancyToApplication(int vacancyId, int applicat
    {
        var vacancy = await _vacancyService.Get(vacancyId);

        if (vacancy == null)
        {
            throw new Exception( message: "Vacancy not found");
        }

        var application = await _applicationService.Get(applicationId);

        if (application == null)
        {
            throw new Exception( message: "Application not found");
        }

        return new CompareResult()
        {
            JobDepartmentMatch = application.JobDepartment == vacancy.JobDepartment,
```

Рисунок 3.10 CompareService (рисунок виконаний самостійно)

Для того щоб відображати результат котрий повертає CompareService, було розроблено по котрому клієнтська частина відмальовує різницю в здібностях та виводить їх у спеціальне модальне вікно(див. рис. 3.11).

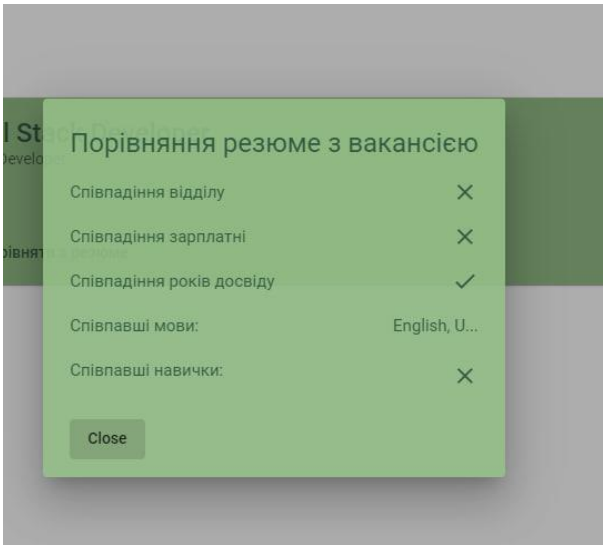


Рисунок 3.11 UI Part (рисунок виконаний самостійно)

4 ОПИС ПРИЙНЯТИХ ПРОГРАМНИХ РІШЕНЬ

4.1 Azure DevOps

Azure DevOps має багато переваг, які роблять його привабливим вибором для управління процесом розробки програмного забезпечення. По-перше, це інтегрована платформа, яка об'єднує різні інструменти та сервіси в одному місці. Вона забезпечує безперервну інтеграцію (CI) та безперервну доставку (CD), що дозволяє командам ефективно автоматизувати збірку, тестування та розгортання додатків.

Azure DevOps забезпечує масштабованість, що дозволяє легко адаптуватися до потреб різних проектів, незалежно від їх розміру. Платформа підтримує як малі команди з кількома розробниками, так і великі організації з тисячами співробітників. Крім того, вона підтримує різні мови програмування, середовища розробки та платформи, що робить її універсальним інструментом для розробників.

Безпека та відповідність є ще однією важливою перевагою Azure DevOps. Платформа забезпечує надійний захист даних, включаючи шифрування та автентифікацію, що гарантує безпеку розробки та розгортання додатків. Azure DevOps також відповідає багатьом міжнародним стандартам і регуляторним вимогам, що важливо для організацій, які працюють у сфері, де потрібна висока відповідність нормативам.

Інтеграція з іншими сервісами Microsoft, такими як Azure, GitHub та Visual Studio, забезпечує ще більшу гнучкість і можливості для розробників. Azure DevOps дозволяє легко інтегруватися з різними інструментами та сервісами, що вже використовуються в організації, що робить перехід на нову платформу плавним та безболісним.

Командна співпраця значно покращується завдяки можливостям Azure DevOps. Платформа забезпечує централізоване управління проектами, дозволяючи командам ефективно планувати, відстежувати та керувати своїми завданнями. Інструменти для управління версіями коду, відстеження помилок та управління релізами дозволяють забезпечити прозорість та злагоджену роботу всіх учасників проекту.

Azure DevOps також відзначається високою доступністю та надійністю. Сервіс забезпечує безперебійний доступ до інструментів та даних, мінімізуючи ризик простою. Крім того, він пропонує широкий спектр аналітичних інструментів, що

дозволяють відстежувати продуктивність команди, виявляти вузькі місця та оптимізувати процес розробки.

Таким чином, Azure DevOps є потужним, масштабованим та універсальним інструментом для управління процесом розробки програмного забезпечення. Він забезпечує інтеграцію різних інструментів, високу безпеку, підтримку командної співпраці, надійність та можливість легко адаптуватися до потреб проекту, що робить його оптимальним вибором для сучасних команд розробників.

4.2 Azure

Вибір Azure як платформи для хмарних обчислень має багато переваг, що роблять його привабливим для підприємств різного розміру і сфер діяльності. Перш за все, Azure забезпечує високу масштабованість, що дозволяє компаніям легко адаптувати свої ресурси відповідно до змінних потреб. Це означає, що ви можете швидко збільшувати або зменшувати обчислювальні потужності, зберігання даних та інші ресурси в залежності від поточних вимог бізнесу.

Azure пропонує широкий спектр послуг та інструментів, включаючи віртуальні машини, бази даних, аналітичні сервіси, штучний інтелект та машинне навчання, що дозволяє створювати комплексні рішення. Ця універсальність дозволяє підприємствам використовувати одну платформу для вирішення різних задач, що підвищує ефективність і зменшує витрати на управління інфраструктурою.

Однією з ключових переваг Azure є його інтеграція з іншими продуктами та сервісами Microsoft, такими як Office 365, Dynamics 365 та Power BI. Це забезпечує безперебійний робочий процес та підвищує продуктивність, оскільки всі інструменти взаємодіють між собою без проблем. Крім того, Azure підтримує різні операційні системи, мови програмування, фреймворки та бази даних, що забезпечує високу гнучкість для розробників.

Azure забезпечує високий рівень безпеки та відповідності, що є критично важливим для підприємств, які працюють з конфіденційними даними. Платформа пропонує широкий спектр засобів захисту, включаючи шифрування, багатофакторну автентифікацію та засоби для моніторингу безпеки. Azure також відповідає багатьом міжнародним стандартам і регуляторним вимогам, що гарантує безпеку та відповідність даних.

Ще однією важливою перевагою Azure є його глобальна доступність. Мережа центрів обробки даних, розташованих по всьому світу, забезпечує високу продуктивність і мінімальну затримку для користувачів, де б вони не знаходилися. Це особливо важливо для компаній з міжнародною присутністю, які потребують надійного доступу до своїх додатків та даних.

Azure також забезпечує надійність та високу доступність своїх сервісів. Завдяки розподіленій архітектурі та вбудованим засобам для резервного копіювання та відновлення, Azure гарантує мінімальний час простою та збереження даних навіть у випадку збоїв. Крім того, платформа пропонує інструменти для управління та моніторингу ресурсів, що дозволяє швидко виявляти і виправляти проблеми.

Загалом, вибір Azure як хмарної платформи забезпечує масштабованість, універсальність, інтеграцію з іншими продуктами Microsoft, високий рівень безпеки та відповідності, глобальну доступність, надійність та високу продуктивність. Ці переваги роблять Azure оптимальним вибором для сучасних підприємств, які прагнуть підвищити ефективність та безпеку своїх хмарних рішень.

4.3 Azure Web App Service

Azure Web App Service має багато переваг, які роблять його привабливим вибором для розгортання та управління веб-застосунками. Перш за все, це повністю керована платформа як послуга (PaaS), що звільняє розробників від необхідності керувати інфраструктурою. Це дозволяє зосередитися на створенні та вдосконаленні застосунків, залишаючи управління серверами, масштабування та інші операційні завдання на платформу Azure.

Однією з головних переваг Azure Web App Service є його здатність автоматично масштабувати застосунки в залежності від навантаження. Це означає, що ваша веб-програма може обробляти пікові навантаження без переривань і забезпечувати високу продуктивність для користувачів. Платформа підтримує автоматичне масштабування в залежності від попиту, що дозволяє знижувати витрати, оскільки ви платите лише за ті ресурси, які фактично використовуєте.

Azure Web App Service забезпечує високу надійність та доступність завдяки вбудованим можливостям для автоматичного відновлення та резервного копіювання. Це гарантує, що ваші застосунки залишаються доступними навіть у випадку збоїв або

непередбачуваних ситуацій. Крім того, платформа пропонує вбудовані засоби для моніторингу та логування, що дозволяє в реальному часі відстежувати стан застосунків та швидко виявляти і виправляти проблеми.

Інтеграція з іншими сервісами Azure, такими як Azure SQL Database, Azure Storage та Azure Active Directory, забезпечує легкість створення комплексних рішень. Це дозволяє використовувати різні можливості хмарної платформи Azure для зберігання даних, аутентифікації користувачів, обробки даних та інших завдань.

Azure Web App Service підтримує безліч мов програмування і фреймворків, таких як .NET, Java, PHP, Node.js, Python, Ruby, і багато інших. Це робить платформу універсальною і придатною для розробників з різними технічними уподобаннями. Крім того, платформа забезпечує безперервну інтеграцію та доставку (CI/CD) з популярними інструментами, такими як GitHub, Bitbucket та Azure DevOps, що дозволяє автоматизувати процес розгортання і оновлення застосунків.

Безпека є ще однією важливою перевагою Azure Web App Service. Платформа забезпечує вбудовані можливості для захисту веб-застосунків, включаючи підтримку SSL/TLS, автентифікацію за допомогою Azure Active Directory, а також інші засоби для захисту даних та доступу. Azure також відповідає багатьом міжнародним стандартам безпеки та регуляторним вимогам, що забезпечує високий рівень захисту даних.

Нарешті, Azure Web App Service пропонує зручну систему управління та розгортання веб-застосунків через портал Azure, інтерфейси командного рядка (CLI) та API. Це дозволяє легко керувати вашими застосунками, виконувати розгортання, оновлення та моніторинг з будь-якого місця і в будь-який час.

Таким чином, вибір Azure Web App Service забезпечує автоматичне масштабування, високу надійність та доступність, інтеграцію з іншими сервісами Azure, підтримку різних мов програмування, вбудовані засоби безпеки та зручне управління. Ці переваги роблять Azure Web App Service оптимальним вибором для розробників, які прагнуть створювати надійні, масштабовані та безпечні веб-застосунки.

5 ТЕСТУВАННЯ РОЗРОБЛЕННОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Тестування програмного забезпечення – це процес оцінки та перевірки, що програма або система працює належним чином відповідно до визначених вимог і очікувань. Це важливий етап розробки, який включає виявлення помилок, перевірку функціональності, продуктивності, безпеки та інших аспектів програми. У сфері DevOps тестування програмного забезпечення є надзвичайно важливим, оскільки воно сприяє забезпеченню якості не тільки коду, але й розробленого середовища, пайплайнів та інших налаштувань.

Автоматизоване тестування дозволяє швидко виявляти помилки на ранніх етапах розробки, що зменшує вартість їх виправлення та підвищує загальну якість програмного забезпечення. Тестування є невід'ємною частиною CI/CD пайплайнів, забезпечуючи, що кожна зміна коду автоматично перевіряється перед інтеграцією в основну гілку або розгортанням на продуктивні середовища. У DevOps тестуються не тільки програми, але й інфраструктура та конфігурації середовища, включаючи перевірку правильності налаштувань серверів, контейнерів, мереж та інших компонентів, щоб забезпечити їх відповідність вимогам.

Автоматизоване тестування скорочує час на ручну перевірку і дозволяє швидше впроваджувати зміни, що особливо важливо в DevOps, де швидкість релізів є критичною. Тестування на вразливості та інші безпекові аспекти допомагає виявляти потенційні загрози та забезпечувати захист додатків та даних. Регресійне тестування гарантує, що нові зміни не порушують вже існуючий функціонал, підтримуючи стабільність додатку.

5.1 Build Pipeline

Build Pipeline має кілька важливих позитивних сторін, які сприяють покращенню процесу розробки програмного забезпечення. Він автоматизує багато рутинних завдань, що дозволяє команді розробників зосередитися на створенні функціональних можливостей і покращенні якості коду. Однією з головних переваг є підвищення продуктивності, оскільки автоматизація зборки і тестування значно скорочує час на ручні процеси. Завдяки цьому, розробники можуть швидше виявляти помилки та виправляти їх, що знижує ризик затримок і проблем під час релізу.

Build Pipeline забезпечує стабільність і передбачуваність процесу розробки. Автоматизовані процеси дозволяють точно відтворювати кожну стадію зборки, тестування і розгортання, що зменшує ймовірність помилок, спричинених людським фактором. Це також підвищує впевненість у якості коду, оскільки кожна зміна проходить через стандартизовані тести та перевірки.

Ще однією важливою перевагою є безперервна інтеграція і доставка (CI/CD). Build Pipeline дозволяє інтегрувати зміни в основну гілку коду і автоматично виконувати збірку, тестування і розгортання, що прискорює цикл розробки і випуску нових версій продукту. Це зменшує час між внесенням змін і їх доставкою до кінцевих користувачів, що підвищує гнучкість і конкурентоспроможність компанії.

Автоматизоване тестування в рамках Build Pipeline підвищує якість програмного забезпечення. Тести виконуються автоматично при кожній збірці, що дозволяє швидко виявляти регресії і нові помилки. Це забезпечує високу якість коду і стабільність продукту на всіх етапах розробки.

Нарешті, Build Pipeline покращує співпрацю між командами. Оскільки всі процеси автоматизовані і стандартизовані, розробники, тестувальники і операційні команди можуть ефективніше взаємодіяти і обмінюватися інформацією. Це сприяє більш тісній співпраці і швидшому вирішенню проблем, що виникають у процесі розробки.

Таким чином, Build Pipeline забезпечує автоматизацію, підвищення продуктивності, стабільність і передбачуваність процесу розробки, сприяє безперервній інтеграції і доставці, покращує якість програмного забезпечення та співпрацю між командами.

Саме через це, в середині Azure DevOps було розроблено Build Pipeline та налаштовано спеціальні політики котрі перевіряють валідність сборки програмного забезпечення перед тим як учасник команди зможе оновити основну гілку репозиторію (див. рис. 5.1). Цей пайплайн автоматизує процес зборки та тестування .NET-застосунків у середовищі Azure DevOps. Він встановлює необхідні інструменти, відновлює залежності, збирає проект з відповідними параметрами, а також виконує тести для забезпечення якості коду.

Цей пайплайн Azure DevOps автоматизує процес зборки, тестування та підготовки артефактів для публікації .NET-застосунку. Він використовує останній

образ Windows для створення необхідного середовища. Визначаються змінні для пайплайну: шлях до файлу рішення (.sln), платформа зборки (Any CPU) та конфігурація зборки (Release). Першим кроком є встановлення інструменту NuGet, необхідного для управління пакетами у .NET-проектах. Далі виконується відновлення всіх необхідних NuGet-пакетів для проекту на основі файлу рішення.

```yaml
pool:
  vmImage: 'windows-latest'

variables:
  solution: '**/*.sln'
  buildPlatform: 'Any CPU'
  buildConfiguration: 'Release'

steps:
Settings
- task: NuGetToolInstaller@1

Settings
- task: NuGetCommand@2
  inputs:
    restoreSolution: '$(solution)'

Settings
- task: VSBuild@1
  inputs:
    solution: '$(solution)'
    msbuildArgs: '/p:DeployOnBuild=true /p:WebPublishMethod=Package /p:PackageAsSingleFile=true /p:
    platform: '$(buildPlatform)'
    configuration: '$(buildConfiguration)'

Settings
- task: VSTest@2
  inputs:
    platform: '$(buildPlatform)'
    configuration: '$(buildConfiguration)'
```

Рисунок 5.1 RecruitXpert CI Pipeline(рисунок виконаний самостійно)

Після цього здійснюється збірка проекту за допомогою Visual Studio Build Task (VSBuild). Вказані аргументи msbuild забезпечують збірку з можливістю розгортання, вказівку на метод публікації як пакет, пакування проекту в один файл, пропуск невірних конфігурацій та збереження пакету у вказаному місці.

Використання цього пайплайну дозволяє забезпечити надійну та ефективну автоматизацію процесу розробки та підготовки артефактів для подальшого розгортання.

У разі якщо пайплайн виконався успішно, то ми побачимо успішний результат та отримаємо змогу провести merge в основну гілку (див. рис. 5.2)

Рисунок 5.2 RecruitXpert CI Pipeline Success(рисунок виконаний самостійно)

У разі якщо пайплайн виконався з помилками, то ми побачимо що він провалився під чав виконання (див. рис. 5.3) і нам треба буде розібратися в чому саме виникла проблема під час його прогону(див. рис. 5.4)

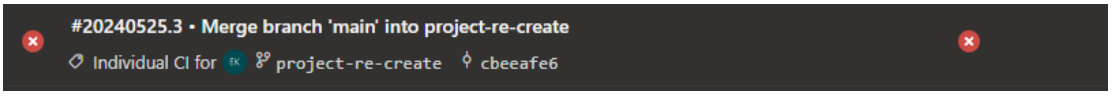


Рисунок 5.3 RecruitXpert CI Pipeline Fail(рисунок виконаний самостійно)

Для того щоб подивитися що саме виклика проблеми, нам треба зайти саме в цей запуск нашого пайплайну натиснувши ліву кнопку миші, та подивитися список помилок (див. рис. 5.4). В залежності від ситуації, можливо також ще треба буде зайти в середину самої послідовності виконання команд щоб побачити яка проблема виникла.

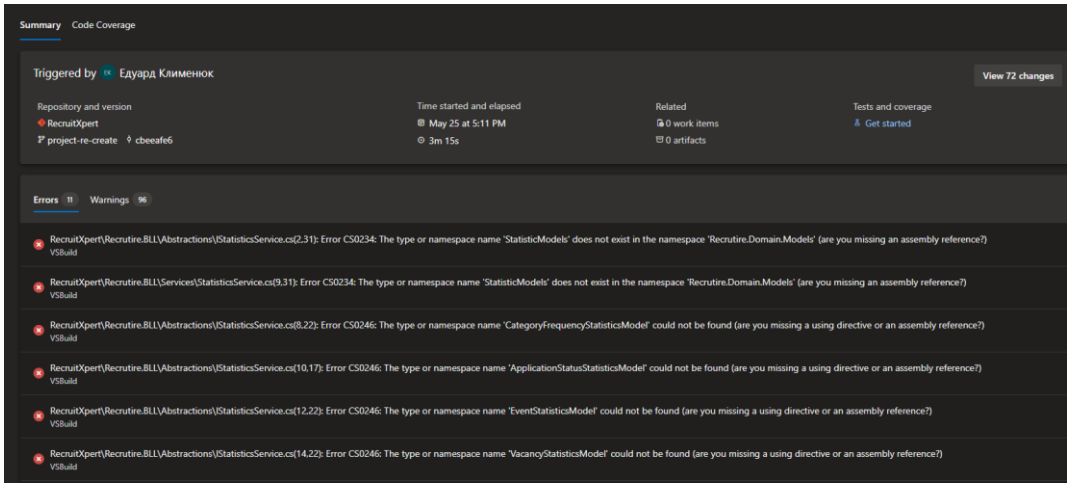


Рисунок 5.4 RecruitXpert CI Pipeline Failed Details(рисунок виконаний самостійно)

Завдяки цьому розробники в команді можуть реагувати на помилки котрі вони могли допустити під час розробки та виправляти їх. Це запобігає можливим конфліктам, що в свою чергу може призвести до непередбачуваних наслідків.

ВИСНОВКИ

У рамках цієї дипломної роботи було досліджено та впроваджено автоматизацію онлайн-рекрутингу, використовуючи сучасні хмарні сервіси Azure. Основну увагу було приділено інтеграції та використанню Azure DevOps для безперервної інтеграції та доставки (CI/CD), що дозволило значно оптимізувати процес розробки, тестування та розгортання додатків.

Azure Web App Service забезпечив швидке і безпечне середовище для роботи веб-додатку, дозволяючи зосередитися на розробці функціоналу, не турбуючись про інфраструктуру. Azure Container Registry використовувався для зберігання та управління контейнерними образами, що значно спростило процес розгортання та підтримки додатку. Завдяки Azure SQL Server було забезпечено надійне та масштабоване зберігання даних, що є критично важливим для роботи сучасних рекрутингових систем.

Azure Storage Account забезпечив безпечне і доступне рішення для зберігання великих обсягів даних, включаючи різноманітні об'єкти та файли, необхідні для роботи додатку. Використання Azure Managed Identity дозволило безпечно та ефективно керувати аутентифікацією додатків і служб, що взаємодіють з іншими ресурсами Azure, усуваючи потребу в управлінні обліковими даними вручну.

Інтеграція всіх цих інструментів та сервісів Azure дозволила досягти високого рівня ефективності та гнучкості в управлінні процесами рекрутингу. Автоматизація значно скоротила час, необхідний для виконання рутинних завдань, підвищила точність і надійність обробки даних, а також забезпечила можливість масштабування системи відповідно до зростаючих потреб бізнесу.

Таким чином, використання Azure DevOps у поєднанні з іншими хмарними сервісами Azure, такими як Web App Service, Container Registry, Azure SQL Server, Storage Account та Azure Managed Identity, показало себе як ефективне та перспективне рішення для автоматизації онлайн-рекрутингу. Це дозволяє значно підвищити продуктивність, якість роботи та забезпечити високий рівень безпеки у сфері підбору персоналу. Автоматизація процесів рекрутингу сприяє покращенню загальної ефективності бізнесу, роблячи його більш конкурентоспроможним на ринку.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. What is DevOps? [Електронний ресурс] – URL: https://about.gitlab.com/topics/devops/ (дата звернення: 07.06.2024)

2. What is Azure. [Електронний ресурс] – URL: https://azure.microsoft.com/en-us/resources/cloud-computing-dictionary/what-is-azure (дата звернення: 07.06.2024)

3. Getting Started with AWS [Електронний ресурс] – URL: https://aws.amazon.com/getting-started/?nc1=h_ls (дата звернення: 07.06.2024)

4. DevOps Architecture [Електронний ресурс] – URL: https://www.javatpoint.com/devops-architecture (дата звернення: 18.05.2022)

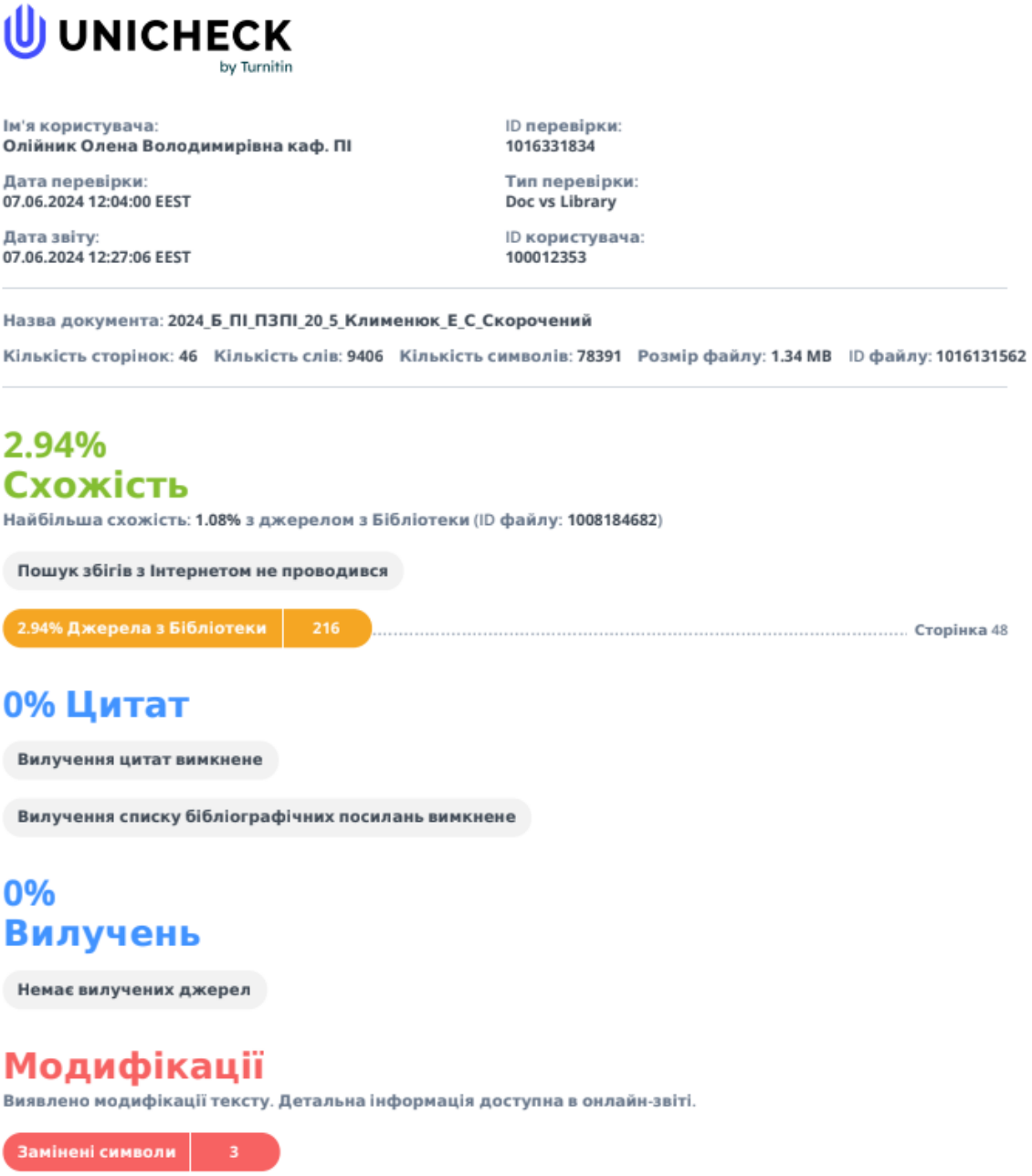
5. Jet Brains WebStorm [Електронний ресурс] – URL: https://www.jetbrains.com/webstorm/ (дата звернення: 12.05.2024).

6. Azure DevOps Repository [Електронний ресурс] – URL: https://azure.microsoft.com/en-us/products/devops/repos (дата звернення: 12.05.2024).

7. Azure Pipelines [Електронний ресурс] – URL: https://azure.microsoft.com/en-us/products/devops/pipelines (дата звернення: 12.05.2024).

ДОДАТОК А

Звіт результатів перевірки на унікальність тексту в базі ХНУРЕ



UNICHECK
by Turnitin

Ім'я користувача:
Олійник Олена Володимирівна каф. ПІ

ID перевірки:
1016331834

Дата перевірки:
07.06.2024 12:04:00 EEST

Тип перевірки:
Doc vs Library

Дата звіту:
07.06.2024 12:27:06 EEST

ID користувача:
100012353

Назва документа: 2024_Б_ПІ_ПЗПІ_20_5_Клименюк_Е_С_Скорочений

Кількість сторінок: 46 Кількість слів: 9406 Кількість символів: 78391 Розмір файлу: 1.34 MB ID файлу: 1016131562

2.94%
Схожість

Найбільша схожість: 1.08% з джерелом з Бібліотеки (ID файлу: 1008184682)

Пошук збігів з Інтернетом не проводився

2.94% Джерела з Бібліотеки 216 Сторінка 48

0% Цитат

Вилучення цитат вимкнене

Вилучення списку бібліографічних посилань вимкнене

0%
Вилучень

Немає вилучених джерел

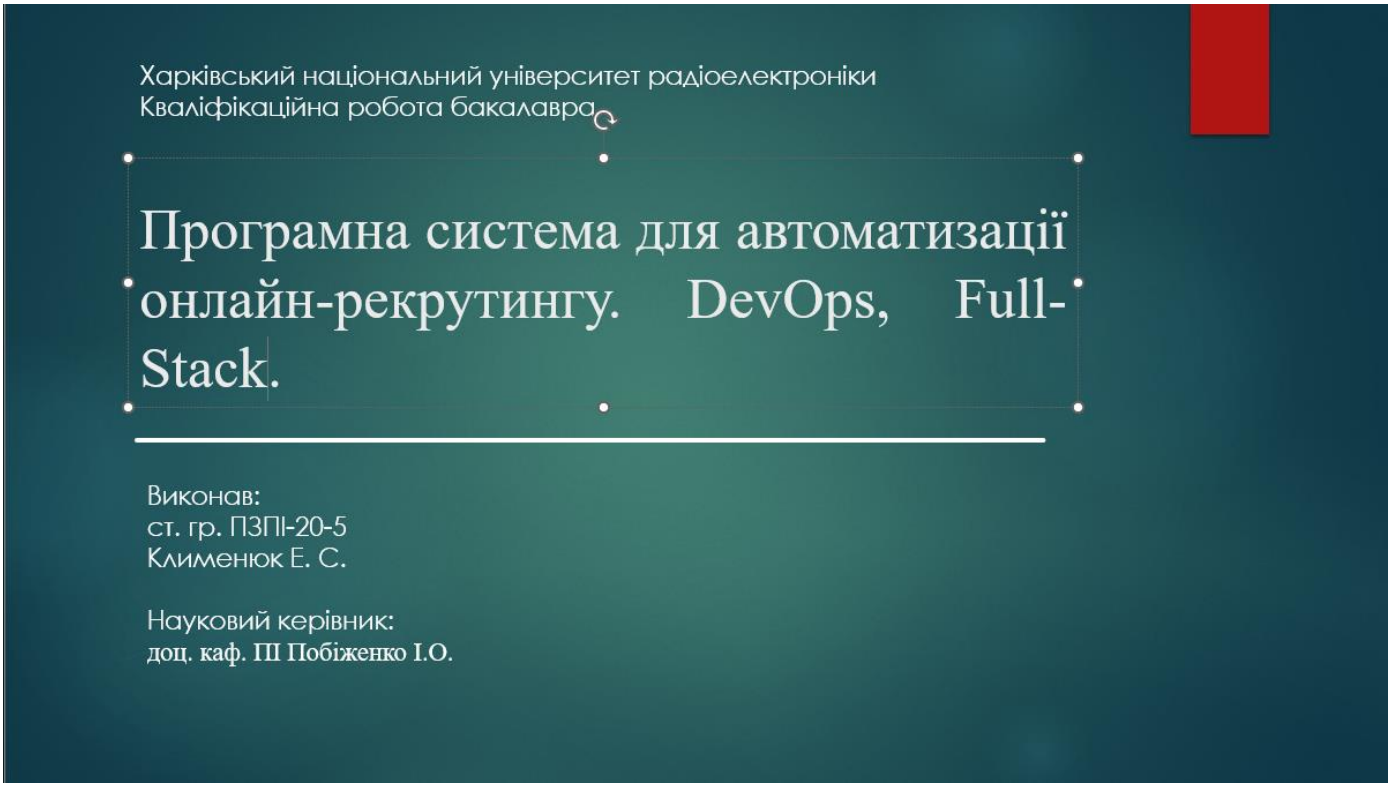
Модифікації

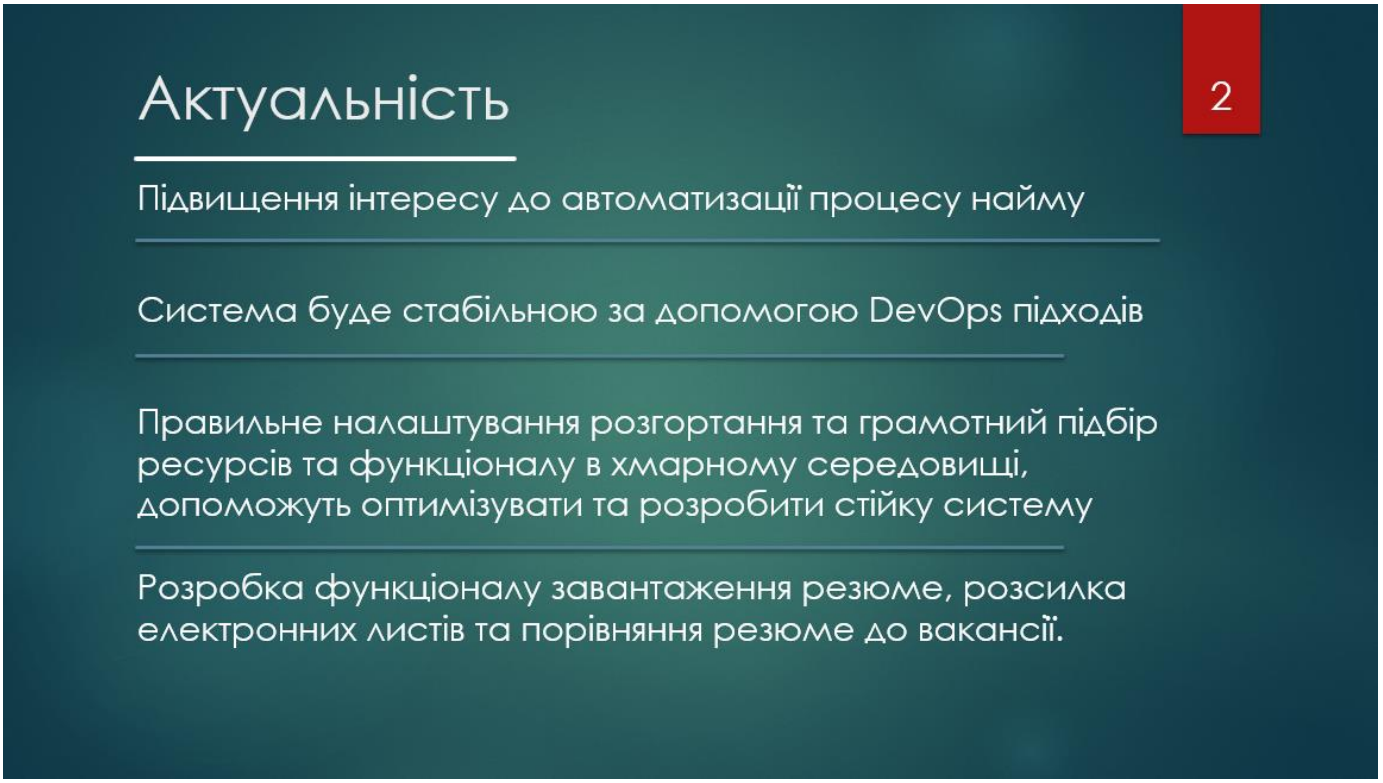
Виявлено модифікації тексту. Детальна інформація доступна в онлайн-звіті.

Замінені символи 3

ДОДАТОК Б

Слайди презентації





Об'єкт роботи та Мета роботи

3

Об'єкт роботи - програмна система для автоматизації онлайн-рекрутингу, що включає в себе розгортання додатку на платформі Azure та налаштування автоматичного деплою.

Мета роботи - спрямована на поліпшення ефективності та організації процесу найму персоналу, автоматизація процесів та покращення використання ресурсів за допомогою сучасних технологій та платформ.

Постановка задачі

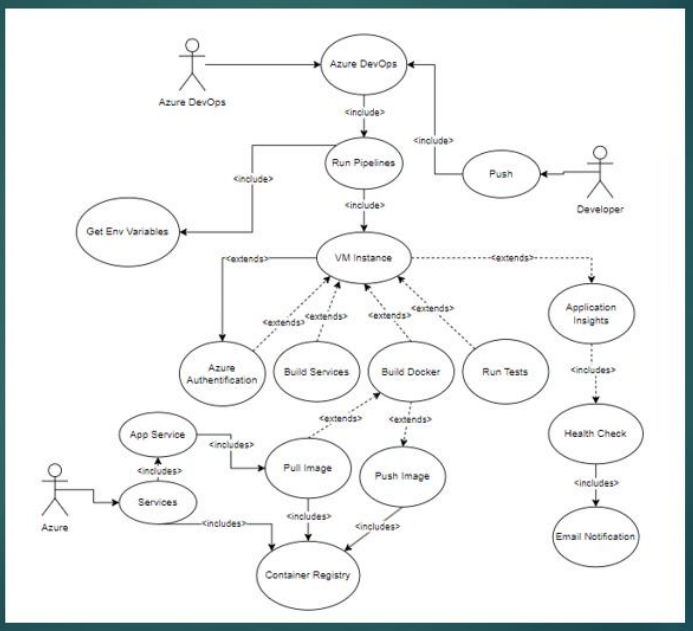
4

Сервіс повинен відповідати таким вимогам:

- автоматизоване розгортання на платформі Azure;
- моніторинг та логування;
- неперервну інтеграцію та доставку (CI/CD);
- тестування;
- завантаження резюме;
- розсилка електронних листів;
- порівняння резюме до вакансії;

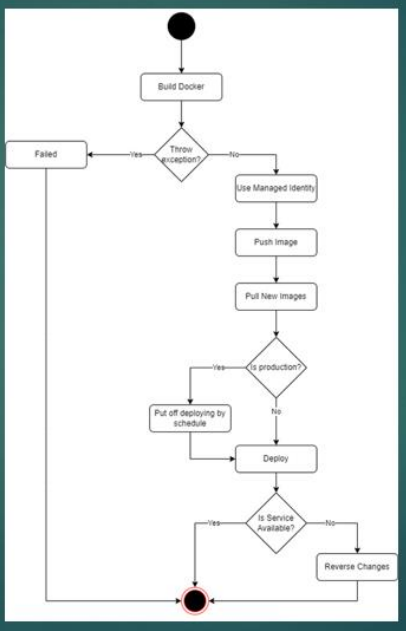
Use Case Diagram

5



Алгоритм деплою Azure

6



7

Чому Azure

Microsoft Azure – це хмарна платформа, яка надає широкий спектр послуг для обчислень, зберігання даних, аналітики, штучного інтелекту та розробки додатків, дозволяючи бізнесам і розробникам створювати, розгортати та керувати додатками через глобальну мережу дата-центрів Microsoft.

Переваги:

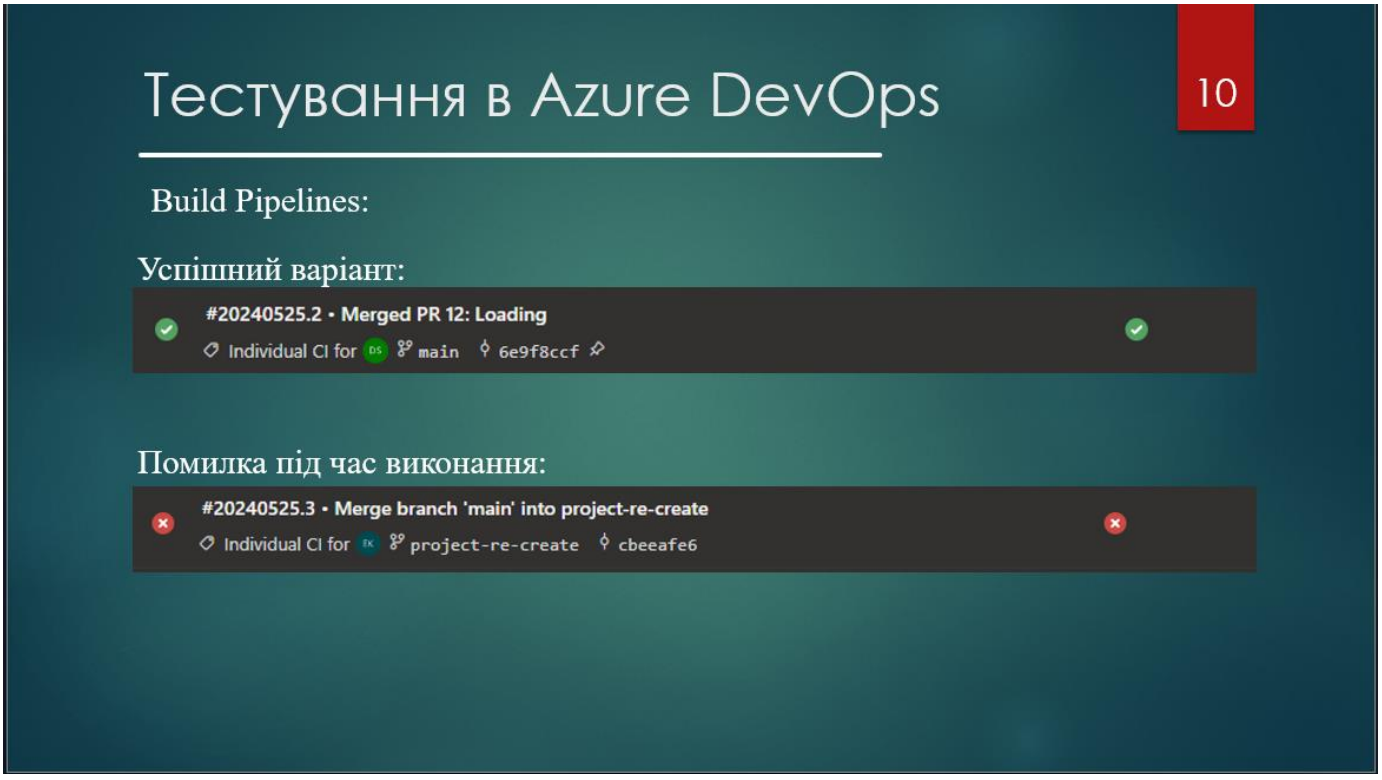
- масштабованість;
- безпеку;
- універсальність та надійність;

8

HangFire

HangFire є багатофункціональною бібліотекою для .NET, що дозволяє виконувати фонові завдання і планувати їх у додатках.


Технології
9
mongoDB
Microsoft Azure
Entity Framework
Core
PostgreSQL


Тестування в Azure DevOps
10
Build Pipelines:
Успішний варіант:
#20240525.2 • Merged PR 12: Loading
Individual CI for main 6e9f8ccf
Помилка під час виконання:
#20240525.3 • Merge branch 'main' into project-re-create
Individual CI for project-re-create cbeeafe6

Апробація

11

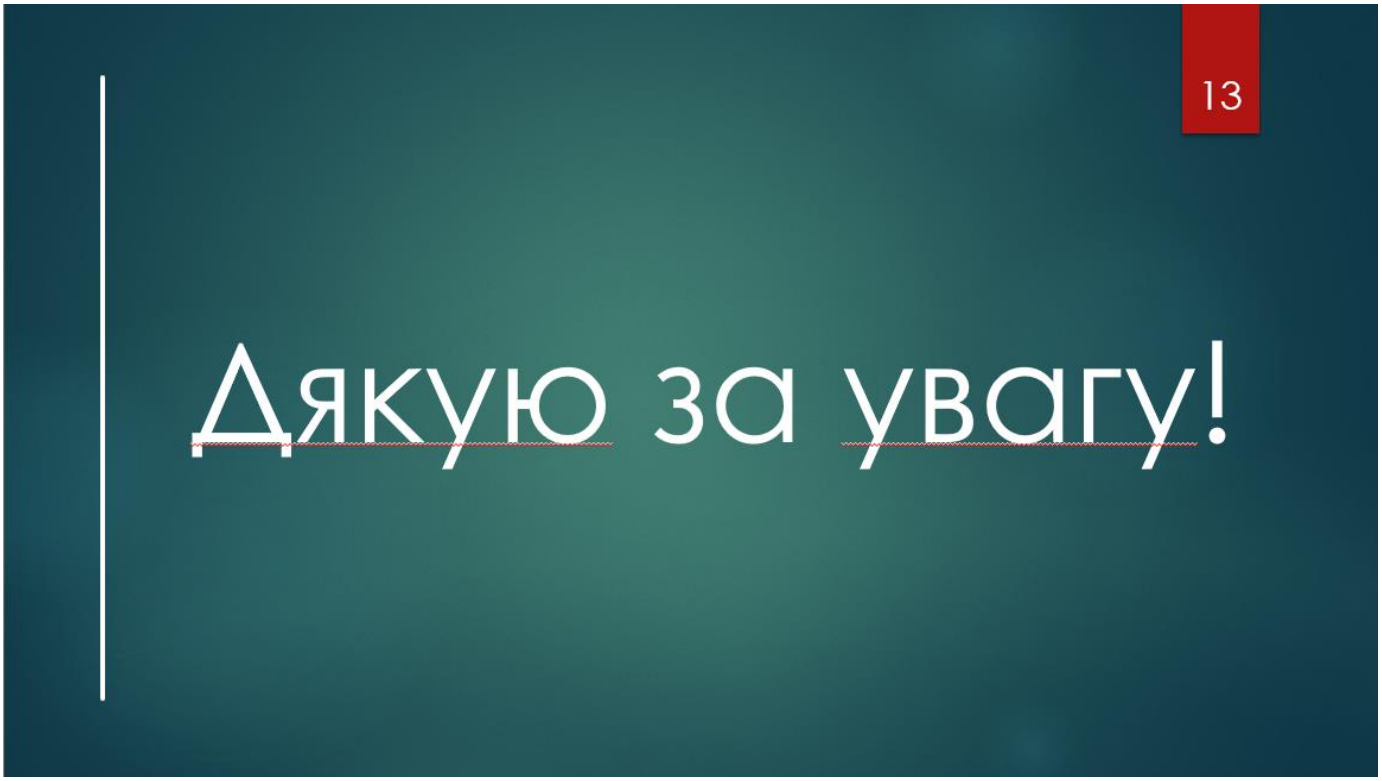
Сертифікат учасника конференції «Print, Multimedia and Web» за доповідь «Аналіз трендів у сфері рекрутингу та їх відображення в функціональних можливостях додатків»

Висновки

12

В результаті роботи було:

- Було визначено мету теми та сформульовані завдання, що стоять перед мною.
- Проаналізовано сучасні підходи в розробці
- Встановлено технології та бібліотеки, що будуть використовуватись.
- Налаштоване та розгорнуте середовище в Azure
- Налаштоване CI\CD


13
Дякую за увагу!

ДОДАТОК В

СПЕЦИФІКАЦІЯ ВИМОГ ДО ПРОГРАМНОГО ПРОДУКТУ

Міністерство освіти і науки України
Харківський національний університет радіоелектроніки
Факультет комп’ютерних наук
Кафедра програмної інжеренії

СПЕЦИФІКАЦІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Програмна система для автоматизації онлайн-рекрутингу

Студент гр. ПЗПІ-20-5 Клименюк Едуард Сергійович
Студент гр. ПЗПІ-20-4 Харченко Поліна Дмитрівна
Студент гр. ПЗПІ-20-4 Стрюкова Дар’я В’ячеславівна

Харків
2024 р.

ЗМІСТ

1 ВСТУП

1.1 Огляд продукту

Програмна система для автоматизації онлайн-рекрутингу є потужним інструментом для компаній, які займаються наймом персоналу. Цей продукт надає комплекс функцій, які сприяють ефективному керуванню процесом рекрутингу та спрощують взаємодію з кандидатами.

Система забезпечує безпеку та контроль доступу за допомогою системи аутентифікації та авторизації. Це гарантує, що доступ до системи мають лише авторизовані користувачі, такі як рекрутери та адміністратори, забезпечуючи конфіденційність даних.

Адміністратори можуть легко керувати користувачами, створювати нові облікові записи, надавати різні рівні доступу та редагувати існуючі профілі. Це дозволяє ефективно організовувати робочий процес та розподіляти обов'язки серед персоналу.

Система дозволяє створювати та редагувати вакансії, розміщувати оголошення про вакансії та отримувати аплікації від кандидатів. Користувачі можуть переглядати деталі вакансій, такі як опис посади, вимоги до кандидатів та умови праці, а також подавати свої резюме. Це спрощує процес рекрутингу та дозволяє рекрутерам ефективно відбирати потенційних кандидатів.

Управління кандидатами є ще одним важливим аспектом продукту. Рекрутери можуть вести базу даних кандидатів, відстежувати їх статуси, проводити співбесіди та приймати рішення щодо найняття.

Ця програмна система сприяє покращенню ефективності процесу рекрутингу, зменшенню часу на пошук та відбір кандидатів, а також підвищенню якості найму.

1.2 Мета

Система для автоматизації онлайн-рекрутингу спрямована на створення зручного та централізованого середовища для організацій, які здійснюють найм персоналу. Головною метою цієї системи є надання керівникам зручного і ефективного інструменту для планування та координації найму персоналу, забезпечуючи їх можливістю організовано проводити відбір кандидатів та вести

облік рекрутингових процесів.

Крім того, система спрощує комунікацію між рекрутерами та кандидатами, надаючи зручний механізм обміну інформацією. Вона автоматично повідомляє про нові вакансії та статуси рекрутингових процесів, що в свою чергу дозволяє кандидату швидко реагувати на зміни та подаватися на нові вакансії.

Загалом, ця програмна система спрямована на поліпшення ефективності та організації процесу найму персоналу, надаючи зручні інструменти для планування, координації, відстеження та звітності щодо вакансій та кандидатів.

1.3 Межі

Програмна система спрямована на забезпечення ефективності та швидкості процесів рекрутингу, зокрема шляхом автоматизації рутинних завдань та зменшення людського втручання.

Цілі програмного продукту включають зменшення часу, потрібного для створення вакансій і подачі заявок, підвищити ефективність процесу найму.

Вона має надавати зручний API для взаємодії з клієнтською частиною додатку та має бути здатна обробляти запити від багатьох користувачів одночасно.

Однак, у системі є обмеження, такі як ресурси сервера (пам'ять та потужність процесора), які впливають на її продуктивність та масштабованість. Це може уповільнити обробку запитів та знизити доступність для користувачів. Також існують обмеження на кількість користувачів та ролей, що можуть бути створені. Додатково, система повинна забезпечувати безпеку даних, використовуючи шифрування та інші методи захисту, щоб убезпечити інформацію від несанкціонованого доступу.

1.4 Означення та абревіатури

Frontend – це частина програмної системи, що відповідає за відображення інтерфейсу користувача та взаємодію з ним через браузер або інші засоби. Вона включає в себе розробку та реалізацію веб-сторінок, компонентів і елементів керування, які користувач може бачити та взаємодіяти з ними.

Backend – це складова частина програмного забезпечення, яка відповідає за

обробку даних та логічні операції на серверному рівні. Вона забезпечує взаємодію з базою даних, виконує процеси автентифікації, авторизації та управління бізнес-логікою системи.

API – це скорочення від Application Programming Interface. Це набір правил і протоколів, що визначає, як програми або компоненти системи можуть взаємодіяти між собою. API визначає доступні функції та методи комунікації між різними складовими програмної системи.

ORM – це абревіатура від Object-Relational Mapping, що означає технологію, яка дозволяє взаємодіяти з базою даних у вигляді об'єктів програми. Вона перетворює дані з реляційної бази даних у об'єкти, що можуть бути використані в програмному коді, та надає зручний спосіб управління даними.

HTTP – це протокол передачі гіпертекстових даних через мережу, що використовується для комунікації між клієнтами та серверами в Інтернеті.

REST – підхід до архітектури мережевих протоколів, які надають доступ до інформаційних ресурсів.

CRUD – це абревіатура, що складається зі слів Create, Read, Update, Delete (створити, прочитати, оновити, видалити). Це основні операції, які можна виконати з даними в базі даних, включаючи створення нових записів, отримання даних, їх оновлення та видалення.

Entity Framework – це технологія ORM (Object-Relational Mapping), яка дозволяє розробникам працювати з базами даних у формі об'єктів програми. Вона забезпечує зручний спосіб доступу до даних з бази даних, перетворюючи їх на об'єкти, з якими можна взаємодіяти в програмному коді.

Identity – це система аутентифікації та авторизації користувачів в програмній системі. Вона дозволяє контролювати доступ до різних частин системи, перевіряючи ідентифікаційні дані користувача та надаючи йому відповідні права доступу.

HR – це скорочення від Human Resources (кадрові ресурси). Це відділ або функціональна область в організації, що відповідає за управління персоналом. Відділ HR займається наймом та звільненням працівників, адмініструванням персональних даних, розвитком персоналу, а також забезпечує вирішення конфліктів та питань стосовно заробітної плати та соціальних пакетів.

JSON (JavaScript Object Notation) – це легкий, текстовий формат обміну

даними, що базується на синтаксисі JavaScript. Він використовується для передачі структурованих даних між програмами, особливо у веб-розробці. Формат JSON складається з пар ключ-значення, де ключі є рядками, а значення можуть бути будь-якого типу даних: рядки, числа, масиви, об'єкти тощо.

CRM (Customer Relationship Management) – програмне забезпечення для організацій, призначене для автоматизації взаємодії із замовниками (клієнтами).

2 ЗАГАЛЬНИЙ ОПИС

2.1 Перспективи продукту

Останнім часом спостерігається значний інтерес до автоматизації процесу найму та підвищення ефективності управління персоналом. Наш продукт виявляє значний потенціал застосування у різних галузях, включаючи компанії будь-якого масштабу, агентства з кадрового рекрутингу та інші сфери.

Прогнозовані напрямки розвитку нашого продукту включають розширення функціоналу та впровадження нових модулів для відповіді на конкретні потреби користувачів. Наприклад, можливості планування та розподілу ресурсів, система оцінки кандидатів, інтеграція з іншими платформами або соціальними мережами, розширена аналітика та звітність.

Крім того, перспективи включають вихід на міжнародний ранок та масштабування продукту для задоволення потреб різних компаній та організацій у різних країнах. Адаптація продукту до міжнародних стандартів та норм у сфері рекрутингу відкриє нові можливості для його впровадження на глобальному рівні.

Загалом, наш продукт має значний потенціал для розвитку та відповіді на зростаючі потреби в управлінні процесом найму персоналу.

2.2 Функції продукту

Основні функції цього продукту включають:

FE-1: Дошка відображення процесу найму (аналог Jira)

FE-2: Календар з відображенням онлайн зустрічей та інших подій

FE-3: Google meets, тобто створення онлайн зустрічі з кандидатом

FE-4: Імейли, відправка з системи на пошту, шаблони

FE-5: Розрахунок заробітної плати

FE-6: Резюме кандидата з можливістю прикріпити файли

FE-7: Тести для вакансії

FE-8: Чат з рекрутером

FE-9: Статистика за вакансіями, компаніями, кандидатами, загальна, стан заявки на вакансію

FE-10: Звіт по вакансії

FE-11: Порівняння кандидатів та резюме

FE-12: Генерація документів, наприклад контракти та листи пропозиції на основі даних кандидата

FE-14: Відгуки

FE-15: Підписка на вакансію та розсилка імейлів

FE-16: Релевантність вакансії вимогам

2.3 Характеристики користувачів

Характеристики користувачів програмної системи для керування процесом набору персоналу з back-end реалізацією можуть бути різноманітними, оскільки система взаємодіє з різними типами користувачів. Основні характеристики користувачів включають:

– кандидати на роботу є ключовими користувачами системи. Вони можуть мати різний досвід роботи, освіту та навички, що впливають на їхню придатність для конкретної вакансії;

– HR менеджери відповідають за керування процесом набору персоналу. Вони мають розширені права доступу та можуть створювати, редагувати та видаляти вакансії, керувати кандидатами, аналізувати звітність та виконувати інші адміністративні функції;

– адміністратори системи відповідають за налаштування та підтримку програмного забезпечення. Вони забезпечують безперебійну роботу системи, вирішують технічні проблеми, які можуть виникнути.

Загалом, різноманітність користувачів враховується у розробці системи, щоб кожна категорія могла ефективно використовувати її функціонал для досягнення своїх цілей у процесі найму персоналу.

2.4 Загальні обмеження

У програмній системі для управління рекрутингом з back-end реалізацією існують обмеження, які можуть вплинути на її ефективність та можливості використання.

Продуктивність та масштабованість системи обумовлені фізичними параметрами серверів, такими як кількість пам'яті, потужність обчислювального процесора, а також пропускна здатність мережі. Ці обмеження можуть суттєво вплинути на швидкість та завантаження системи під час обробки великої кількості запитів одночасно.

Забезпечення безпеки є важливою складовою для захисту конфіденційності даних користувачів та запобігання несанкціонованому доступу. Обмеження в цьому плані можуть включати криптографічні вимоги, захист від потенційних атак, встановлення правил доступу до інформації та контроль над правами користувачів.

Система повинна мати можливість масштабуватися для відповіді на зростаючий обсяг користувачів, проектів та завдань. Обмеження масштабованості можуть впливати на продуктивність та швидкодію системи під час інтенсивного навантаження.

2.5 Припущення й залежності

AS-1: Доступ до Інтернету. Передбачається, що всі користувачі мають доступ до Інтернету для використання системи онлайн-рекрутингу.

AS-2: Відповідність інформації в резюме. Передбачається, що інформація, надана кандидатами у їхніх резюме, є точною і відповідає їхньому досвіду та навичкам.

AS-3**:** Стабільність технічних систем. Передбачається, що технічні системи, необхідні для роботи програмної платформи, будуть працювати стабільно без значних перебоїв.

DP-1: Доступ до бази даних кандидатів. Система залежить від наявності доступу до бази даних з профілями кандидатів для ефективного пошуку та відбору претендентів на вакансії.

DP-2: Інтеграція з існуючими системами рекрутингу. Може бути необхідною

інтеграція з існуючими системами рекрутингу або обліку кандидатів, які вже використовуються компанією.

DP-3: Підтримка технічного персоналу. Для ефективної роботи системи може знадобитися технічна підтримка для моніторингу, усунення неполадок та оновлення програмного забезпечення.

3 КОНКРЕТНІ ВИМОГИ

3.1 Вимоги до зовнішніх інтерфейсів

3.1.1 Інтерфейс користувача

Система повинна мати інтуїтивно зрозумілий та ергономічний інтерфейс, що дозволяє користувачам легко взаємодіяти з програмним забезпеченням. Інтерфейс користувача має бути зручним у використанні та забезпечувати швидкий доступ до всіх основних функцій системи.

Крім того, інтерфейс повинен бути адаптивним для різних типів пристроїв, включаючи комп'ютери, планшети та мобільні девайси.

Інтерфейс користувача представлений у вигляді веб-додатку, залежно від потреб користувачів та доступних платформ. Додаток має послідовну організацію, зрозумілу навігаційну систему та інтуїтивно зрозумілі елементи управління, що спрощують користувачам взаємодію та виконання різноманітних завдань.

3.1.2 Апаратний інтерфейс

Апаратне забезпечення програмної системи для управління рекрутингом з back-end реалізацією складається з необхідних компонентів та засобів, які гарантують безперебійну роботу системи. Оскільки програма зазвичай працює в онлайн-середовищі та доступна через веб-браузер, апаратне забезпечення включає:

– сервери для зберігання та обробки даних. Це можуть бути фізичні сервери, віртуальні сервери або хмарні платформи, які забезпечують необхідні обчислювальні та зберігальні ресурси для ефективної роботи системи;

– мережеві пристрої, такі як маршрутизатори, комутатори та мережеві кабелі, для забезпечення безперервного зв'язку між користувачами та серверами системи через мережу.

3.1.3 Програмний інтерфейс

Програмна система для автоматизації онлайн-рекрутингу впроваджує програмний інтерфейс (API) як ключовий механізм для спільної роботи з іншими додатками та системами. Цей API визначає набір правил та протоколів для взаємодії між різними компонентами програмного забезпечення, що дозволяє зовнішнім системам або розробникам використовувати функціональні можливості та отримувати доступ до даних програмної системи.

У контексті онлайн-рекрутингу, API може включати в себе різноманітні функції, такі як створення та оновлення вакансій, перегляд кандидатів, управління процесом найму та багато іншого. Реалізація такого API у формі RESTful API є популярним підходом, оскільки він базується на використанні HTTP-протоколу, що дозволяє здійснювати просту та стандартизовану взаємодію між клієнтами та сервером.

Однією з переваг RESTful API є його простота та легкість використання. Використання стандартних HTTP-методів, таких як GET, POST, PUT та DELETE, спрощує процес комунікації з системою. Крім того, обмін даними у форматі JSON дозволяє ефективно передавати структуровану інформацію між клієнтом та сервером.

Застосування API у програмній системі для онлайн-рекрутингу відкриває безліч можливостей для інтеграції з іншими системами та сервісами.

Використання API у програмній системі для рекрутингу також дозволяє забезпечити гнучкість та розширюваність системи. Розробники можуть легко додавати нові функції та можливості, не змінюючи основну архітектуру програми, що робить систему більш адаптивною до змін потреб бізнесу та ринкових умов.

У цілому, використання програмного інтерфейсу в програмній системі для автоматизації онлайн-рекрутингу є ключовим фактором для забезпечення її ефективності, гнучкості та інтегрованості з іншими системами.

3.1.4 Комунікаційний протокол

Система повинна підтримувати різні інтерфейси комунікації для забезпечення обміну даними з іншими системами та сервісами. Це може включати в себе

стандартні протоколи передачі даних, такі як HTTP/ HTTPS для взаємодії з веб-серверами, REST для взаємодії з іншими веб-службами, а також можливості інтеграції з поштовими клієнтами. Крім того, система повинна забезпечувати захист даних та аутентифікацію при здійсненні комунікації з іншими системами, щоб забезпечити конфіденційність та цілісність інформації.

3.1.5 Обмеження пам’яті

Обмеження пам'яті є критичним аспектом у розробці програмного забезпечення, особливо в сучасних системах, де ефективне використання ресурсів має вирішальне значення для забезпечення продуктивності та надійності програм.

Під час розробки програмного забезпечення важливо усвідомлювати обмеження пам'яті, які встановлені для різних типів пристроїв та платформ. Наприклад, вбудовані системи можуть мати обмежену кількість доступної пам'яті, тоді як сервери або персональні комп'ютери можуть мати більші ресурси.

Одним з методів управління обмеженням пам'яті є ефективне використання алгоритмів та структур даних. Наприклад, використання компактних представлень даних, які займають менше пам'яті, може допомогти знизити використання ресурсів. Крім того, уникання зайвого копіювання даних та оптимізація роботи з пам'яттю також може покращити продуктивність програми.

Для ефективного управління пам'яттю важливо також враховувати процеси видалення та звільнення пам'яті. Наприклад, у системах, де використання пам'яті є критичним, важливо уникати витоку пам'яті шляхом правильного вивільнення непотрібних ресурсів та використанням механізмів автоматичного управління пам'яттю.

3.2 Атрибути програмного продукту

3.2.1 Надійність

Надійність програмної системи для управління онлайн-рекрутингом є ключовою складовою успішного функціонування. Система має забезпечувати

безперебійну та стабільну роботу, що є критично важливим для ефективного управління процесом рекрутингу та задоволення потреб користувачів.

Надійність означає, що система працює так, як очікується, та обробляє запити користувачів, запобігаючи можливим збоям чи відмовам. Для досягнення цієї мети система має бути забезпечена вбудованим механізмом обробки помилок та винятків, що можуть виникати під час роботи. Це допомагає виявляти та вирішувати проблеми, що впливають на роботу системи, забезпечуючи її надійну та стабільну функціональність.

Окрім цього, важливо мати механізми резервного копіювання, які забезпечують збереження та захист даних системи. Це може включати регулярне резервне копіювання даних, створення резервних копій баз даних та інших важливих ресурсів. Такий підхід дозволяє підтримувати надійність та безпеку даних у будь-який час, що є важливим аспектом для успішної роботи програмної системи з управління рекрутингом.

3.2.2 Доступність

Для забезпечення доступності системи можна використовувати різні технології та стратегії. Одним з ключових методів є використання дублювання та резервування. Це означає створення дублюючих екземплярів системи або компонентів, які автоматично вступають в дію у випадку відмови основного обладнання або програмного забезпечення.

Крім того, для забезпечення доступності можна використовувати системи моніторингу та управління. Такі системи постійно контролюють стан обладнання та програмного забезпечення і сповіщають адміністраторів у випадку виявлення проблем. Додатково, вони можуть автоматично виконувати заходи для відновлення роботи системи або виконувати переведення навантаження на резервні сервери.

Одним з важливих аспектів є розробка системи з урахуванням масштабованості горизонтальної або вертикальної. Горизонтальна масштабованість передбачає розширення системи за рахунок додавання нових серверів або вузлів у мережу, щоб розділити навантаження та забезпечити стабільну роботу при зростанні обсягу даних або користувачів. Вертикальна масштабованість включає у підвищення продуктивності і потужності окремих компонентів системи, наприклад, шляхом

оновлення обладнання або програмного забезпечення.

3.2.3 Безпека

Система має забезпечувати високий рівень безпеки для захисту конфіденційності, цілісності та доступності даних. Це охоплює застосування сучасних методів шифрування для конфіденційної інформації, механізми автентифікації для запобігання несанкціонованому доступу та механізми контролю доступу для обмеження прав доступу.

Система повинна використовувати параметризовані запити і виключити можливість SQL-ін'єкцій та інших атак, для забезпечення безпеки запитів до бази даних через API. Також важливо реалізувати аутентифікацію і авторизацію на рівні API, щоб контролювати доступ до даних та забезпечити їх конфіденційність.

3.2.4 Супроводжуваність

Забезпечення документацією відповідає за збереження знань та інформації про систему, що дозволяє новим членам команди ефективно розібратися в її структурі та функціональності. Моніторинг та логування відіграють ключову роль у виявленні проблем та відслідковуванні причин виникнення помилок, що дозволяє оперативно реагувати на них та уникати критичних збоїв у роботі системи.

Регулярні оновлення та тестування забезпечують актуальність та надійність програмного забезпечення, а також виявлення та виправлення помилок ще до їх впливу на користувачів. Підтримка користувачів є важливою для забезпечення задоволення та впевненості користувачів у якості та ефективності системи.

Автоматизація процесів супроводження дозволяє зменшити час та зусилля, витрачені на ручні операції, а також забезпечує більшу стабільність та ефективність управління системою. Включення цих механізмів у пункт про супроводжуваність дозволяє забезпечити готовність та надійність системи для подальшого функціонування після її впровадження.

3.2.5 Переносність

Система повинна бути легко переносною між різними середовищами та платформами. Це включає здатність запускати систему на різних операційних системах, таких як Windows, macOS та Linux, а також на різних типах апаратного забезпечення. Переносність також означає, що система повинна бути здатною працювати в різних мережевих середовищах із різними конфігураціями мережі та доступу до Інтернету.

Крім того, важливо, щоб система могла легко інтегруватися з іншими додатками та сервісами, що використовуються у сфері рекрутингу, такими як CRM-системи, платформи соціальних мереж та рекрутингові портали, незалежно від їхніх технологій чи платформ.

3.2.6 Продуктивність

У програмній системі для управління онлайн-рекрутингом, важливим аспектом є продуктивність, яка включає в себе оптимізацію коду, масштабування та ефективне використання ресурсів. Написання оптимізованого коду, що забезпечує швидку обробку завдань. Масштабування системи для того, щоб ефективно розподіляти навантаження та збалансовано використовувати ресурси, що сприяє більшій продуктивності та підвищення досвіду користувача при користуванні продуктом.

3.3 Вимоги до бази даних

Забезпечення вимог до бази даних є важливим етапом у процесі розробки програмного забезпечення, оскільки це визначає основні принципи організації, зберігання та доступу до даних, необхідних для ефективної роботи системи.

Перш за все, важливо визначити структуру даних, яка відповідає потребам програми. Це включає в себе створення таблиць, визначення полів та їх типів, а також встановлення зв'язків між таблицями для забезпечення консистентності даних.

Далі, важливо нормалізувати дані, щоб уникнути дублювання та забезпечити їх цілісність. Це допомагає забезпечити ефективність роботи з базою даних та уникнути виникнення проблем при маніпулюванні даними.

Однією з ключових вимог є забезпечення безпеки даних. Це включає в себе встановлення прав доступу до даних, що забезпечує конфіденційність інформації, а також застосування методів шифрування для захисту даних від несанкціонованого

доступу.

Крім того, важливо забезпечити швидкодію бази даних шляхом оптимізації запитів та використання індексів. Це допомагає забезпечити ефективну обробку даних та швидку відповідь на запити користувачів.

Не менш важливою є можливість резервного копіювання та відновлення даних. Це дозволяє відновити інформацію в разі втрати або пошкодження даних, що забезпечує безпеку та надійність роботи системи.

ДОДАТОК Г

Сертифікат учасника конференції «Print, Multimedia and Web» за доповідь «Аналіз трендів у сфері рекрутингу та їх відображення в функціональних можливостях додатків»