

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»**

Факультет інформатики та обчислювальної техніки

(повна назва інституту/факультету)

Інформатики та програмної інженерії

(повна назва кафедри)

«На правах рукопису»
УДК 004.273

До захисту допущено:

В.о. завідувача кафедри

_____ Едуард ЖАРІКОВ

«__» _____ 20__ р.

Магістерська дисертація

на здобуття ступеня магістра

за освітньо-професійною програмою «Інженерія програмного
забезпечення інформаційних систем»

зі спеціальності 121 «Інженерія програмного забезпечення»

на тему: «Архітектурне рішення для програмного забезпечення
обробки оцифрованих зображень за допомогою хмарних технологій»

Виконав (-ла):

студент (-ка) VI курсу, групи ІТ-04мп

Царук Володимир Вікторович _____

Науковий керівник: к.т.н,

доцент

Фіногенов Олексій Дмитрович _____

Рецензент:

д.т.н., доц

Корнага Ярослав Ігорович _____

Засвідчую, що у цій магістерській
дисертації немає запозичень з праць
інших авторів без відповідних
посилань.

Студент (-ка) _____

Київ – 2021 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра інформатики та програмної інженерії

Рівень вищої освіти – другий (магістерський)

Спеціальність – 121 «Інженерія програмного забезпечення»

Освітньо-професійна програма - «Інженерія програмного забезпечення інформаційних систем»

ЗАТВЕРДЖУЮ

В.о. завідувача кафедри

_____ Едуард ЖАРІКОВ

«__» _____ 20__ р.

ЗАВДАННЯ

на магістерську дисертацію студенту

Царуку Володимирі Вікторовичу

1. Тема дисертації: «Архітектурне рішення для програмного забезпечення обробки оцифрованих зображень за допомогою хмарних технологій», науковий керівник дисертації Фіногенов Олексій Дмитрович, доцент кафедри інформатики та програмної інженерії, затверджені наказом по університету від «27» жовтня 2021 р. №3587-с
2. Термін подання студентом дисертації: 11.12.2021
3. Об'єкт дослідження: процес обробки оцифрованих зображень за допомогою хмарних технологій.
4. Вхідні дані: постановка задачі
5. Перелік завдань, які потрібно розробити: аналіз архітектур, аналіз хмарних постачальників, розв'язування математичних задач, проектування архітектури
6. Орієнтовний перелік графічного (ілюстративного) матеріалу:
 - 1) схема архітектури програмного забезпечення;
 - 2) діаграма використання;

- 3) алгоритм обробки зображень на сторонній сервіс;
- 4) лістинг програмного коду;
- 5) перевірка на антиплагіат.

7. Орієнтовний перелік публікацій: Царук В.В., «Кольоризація оцифрованих чорно-білого зображення з використанням згорткових мереж».

8. Консультанти розділів дисертації

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
<i>Графічний</i>	<i>доц. Фіногенов О.Д.</i>		

9. Дата видачі завдання 1.09.2021

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1	<i>Вивчення рекомендованої літератури</i>	<i>01.09.2021</i>	
2	<i>Аналіз існуючих методів розв'язання задачі</i>	<i>10.09.2021</i>	
3	<i>Постановка та формалізація задачі</i>	<i>15.09.2021</i>	
4	<i>Моделювання ПЗ</i>	<i>24.09.2021</i>	
5	<i>Визначення з дата сетами</i>	<i>27.09.2021</i>	
6	<i>Аналіз та реалізація існуючих архітектур і дослідження їх</i>	<i>4.10.2021</i>	
7	<i>Розробка архітектури ПЗ</i>	<i>01.11.2021</i>	
8	<i>Розробка і налагодження програмного забезпечення</i>	<i>08.11.2021</i>	
9	<i>Оформлення пояснювальної записки</i>	<i>20.11.2021</i>	
10	<i>Подання МД на попередній захист</i>	<i>27.11.2021</i>	
11	<i>Подання МД рецензенту</i>	<i>10.12.2021</i>	
12	<i>Подання МД на основний захист</i>	<i>22.12.2021</i>	

Студент

Володимир ЦАРУК

Науковий керівник

Олексій ФІНОГЕНОВ

РЕФЕРАТ

Розмір пояснювальної записки – 86 аркушів, містить 21 ілюстрацію, 32 таблиці, 4 додатка. Архітектурне рішення для програмного забезпечення обробки оцифрованих зображень за допомогою хмарних технологій.

Актуальність теми. На сьогоднішній день зображення є невід’ємною частиною нашого життя, адже з кожним роком в мережі інтернет збільшується кількість використаних зображень та відеоматеріалів. Тому питання попередньої обробки і аналізу фотоматеріалів набуває все більш актуального значення. Багато сучасних систем, особливо ті, які зв’язані з медициною почали перехід на підтримку та генерацію зображень. Дані обставини вимагають від програмного забезпечення витримувати суттєве збільшення обсягу виконуваних робіт без суттєвої втрати швидкодії, можливість роботи, як незалежної системи так і інтеграції в алгоритмічні процеси зовнішніх систем. Спроектвана система повинна забезпечити легкість масштабування та можливість використання сучасних сервісів зокрема хмарних технологій.

Мета дослідження: архітектура програмного забезпечення по обробці оцифрованих зображень на основі використання хмарних технологій.

Для реалізації поставленої мети були **сформульовані наступні завдання, а саме:**

- аналіз сучасних рішень та систем в яких можна використовувати дане програмне забезпечення;
- вибір необхідного функціоналу для роботи з зображеннями, як для систем, які б могли інтегрувати розроблене програмне забезпечення;
- вибір стеку технологій, який базується на основі тенденцій розвитку ринку хмарних технологій, а саме розгляд Google Cloud, Amazon Cloud, Microsoft Azure;
- проектування та реалізація хмарної архітектури по обробці оцифрованих зображень можливістю легкого масштабування;

– аналіз отриманих результатів.

Об’єкт дослідження: процес обробки оцифрованих зображень за допомогою хмарних технологій.

Предмет дослідження: є архітектурні рішення та підходи для перетворення та редагування оцифрованого зображення з використанням хмарних технологій.

Наукова новизна результатів магістерської дисертації полягає в тому, що вдосконалено архітектурне рішення на основі хмарних технологій для побудови програмного забезпечення для обробки оцифрованих фотографій, яке забезпечує можливість збільшення швидкості відправки зображень на сторонній сервіс та може інтегруватися у якості сервісу з іншими системами, що дає змогу побудови складних процесів по обробці зображень.

Практичне значення отриманих результатів полягає в тому, що реалізовано програмне забезпечення з використання хмарних технологій, що містить функції обробки, передачі, зберігання, персоналізації та захисту персональної інформації. Спроектвана система може бути використана в процесах обробки зображень для медичних закладів, аптек, в системах електронної комерції тощо.

Зв’язок з науковими програмами, планами, темами. Робота виконувалась в межах теми «Методи та технології високопродуктивних обчислень та обробки надвеликих масивів даних», державний реєстраційний номер 0117U000924, на кафедрі інформатики та програмної інженерії Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського».

Публікації. «Кольоризація оцифрованих чорно-білого зображення з використанням згорткових мереж» в науковому журналі «Вчені записки Таврійського національного університету імені В. І. Вернадського. Серія: Технічні науки», Том 32 (71) № 6 за 2021.

Ключові слова: ХМАРНІ ТЕХНОЛОГІЇ, ОБРОБКА ЗОБРАЖЕНЬ, AWS WAF, AMAZON ROUTE S3, AWS LAMBDA.

ABSTRACT

Explanatory note size – 86 pages, contains 21 illustrations, 32 tables, 4 applications.

Architectural solution for digital image processing software using cloud technologies.

Topicality. Today, images are an integral part of our lives, as the number of images and videos used on the Internet increases every year. Therefore, the issue of pre-processing and analysis of photographic materials is becoming increasingly important. Many modern systems, especially those related to medicine, have begun to move to support and generate images. These circumstances require the software to withstand a significant increase in the amount of work performed without significant loss of speed, the ability to work as an independent system and integration into the algorithmic processes of external systems. The designed system should provide ease of scaling and the ability to use modern services, including cloud technologies.

The aim of the study. The main target of this master's dissertation is architecture of digital image processing software based on the use of cloud technologies.

Object of research: the process of processing digital images using cloud technologies.

Subject of research: methods of converting and editing a digital image using cloud technologies.

To achieve this goal, the **following tasks** were formulated:

- analysis of modern solutions and systems in which you can use this software;
- selection of the necessary functionality for working with images, as for systems that could integrate the developed software;
- the choice of technology stack, which is based on trends in the cloud technology market, namely the consideration of Google Cloud, Amazon Cloud, Microsoft Azure;
- design and implementation of cloud architecture for processing digital images with the possibility of easy scaling;
- analysis of the obtained results.

The scientific novelty of the results of the master's dissertation is that the improved architectural solution based on cloud technologies for building software for digital photo processing, which provides the ability to increase the speed of sending images to third-party service and can be integrated as a service with other systems. complex image processing processes.

The practical value of the obtained results is that the software for the use of cloud technologies is implemented, which contains the functions of processing, transmission, storage, personalization and protection of personal information. The designed system can be used in image processing processes for medical institutions, pharmacies, e-commerce systems and more.

Relationship with working with scientific programs, plans, topics. Relationship with working with scientific programs, plans, topics: the work was performed under the direction of EPAM SYSTEMS Limited Liability Company. Work was performed at the Department of Informatics and Software Engineering of the National Technical University of Ukraine «Kyiv Polytechnic Institute. Igor Sikorsky» also.

Publications. Publications: «Colorization of digitized black and white images using convolutional networks» was adopted by the scientific journal «Scientific Notes of the Tavriya National University named after VI Vernadsky». Series: Technical Sciences ”, Volume 32 (71) № 6 for 2021.

Keywords: CLOUD TECHNOLOGIES, IMAGE PROCESSING, AWS WAF, AMAZON ROUTE S3, AWS LAMBDA.

ЗМІСТ

ВСТУП	6
1 АНАЛІЗ СИСТЕМ-АНАЛОГІВ ПО ОБРОБЦІ ОЦИФРОВАНИХ ЗОБРАЖЕНЬ	8
1.1 Опис предметної області.....	8
1.2 Аналіз існуючих рішень.....	9
1.2.1 Сервіс Colorize B&W Photos	9
1.2.2 Сервіс Pho.to	12
1.2.3 Сервісу PhotoshopOnline	14
1.2.4 Сервіс «Комп'ютерний зір»	16
1.2.5 Переваги використання хмарних технологій	18
1.3 Висновки до розділу 1	20
2 ПОРІВНЯННЯ ХМАРНИХ СЕРВІСІВ ТА ОГЛЯД ТЕХНОЛОГІЙ ДЛЯ ПОБУДОВИ АРХІТЕКТУРИ	22
2.1 Розгляд провідних хмарних компаній	22
2.2 Переваги та недоліки AWS, Azure та Google.....	23
2.2.1 Переваги та недоліки AWS	24
2.2.2 Переваги та недоліки Microsoft Azure.....	24
2.2.3 Переваги та недоліки Google Cloud Platform.....	25
2.3 Математичне забезпечення.....	28
2.3.1 Поняття шарів згорткової мережі	28
2.3.2 Тренування моделі	32
2.3.3 Ініціалізація моделі	34
2.3.4 Оптимізація виконуваного алгоритму	35

	3
2.4 Висновок до розділу 2	37
3 ПРОЕКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ.....	38
3.1 Опис завдання	38
3.2 Використання Serverless	39
3.3 Розробка програмного забезпечення	40
3.4 Аналіз безпеки даних користувача	45
3.5 Опис таблиць в DynamoDB	47
3.6 Демонстрація програмного забезпечення	50
3.7 Експериментальні дослідження	54
3.8 Оцінка ефективності запропонованого рішення	54
3.9 Висновки по розділу 3.....	58
4 МАРКЕТИНГОВИЙ АНАЛІЗ СТАРТАП-ПРОЄКТУ.....	59
4.1 Опис ідеї проєкту.....	59
4.2 Технологічний аудит ідеї проєкту	61
4.3 Аналіз ринкових можливостей запуску стартап-проєкту	62
4.4 Розроблення ринкової стратегії проєкту.....	74
4.5 Розроблення маркетингової програми стартап-проєкту	77
4.6 Висновки по розділу 4.....	80
ВИСНОВКИ	82
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	84
ДОДАТКИ	
ДОДАТОК А	
ДОДАТОК Б	

ДОДАТОК В

ДОДАТОК Г

ДОДАТОК Д

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

API – це інтерфейс, який надається користувачу для взаємодії та засобів для створення програмного забезпечення.

Amazon Route S3 – це високопродуктивний та легко маштабований хмарний веб-сервіс систем доменних імен або DNS.

AWS – це Amazon Web Services, дана система є найбільш поширеною в світі, це є хмарна платформа з великими можливостями. Вони що забезпечують більше 200 функціональних сервісів для центрів обробки інформації по всій планеті.

AWS WAF – це брандмауер для додатків, основна функція якого захищати додатки або API від розповсюджених мережевих випробувань, здатних впливати на доступність, створювати загрозу безпеці API або додатків.

AWS Lambda – це обчислювальний сервіс, основна мета якого виконувати код без необхідності випускати сервери та керувати ними, створювати логіку маштабування кластера, підтримувати інтеграції подій.

CRM – управління відносинами з клієнтами.

DMS – система управління документами.

ERP – планування ресурсів підприємства.

ВСТУП

Сучасний ринок та користувачі потребують швидкої та безпечної обробки даних. В реальності, якщо користувач протягом тривалого часу почне використовувати програмне забезпечення, яке не дає очікуваних результатів, він почне використовувати інший аналог.

Усі види обробки, зберігання та простий доступ до фотографій стають невід'ємною частиною нашого життя. Багато різноманітних сервісів починають інтегрувати в свої системи роботу з зображенням.

В останні роки в галузі хмарних обчислень з'явилися нові тенденції. Хоча хмарні обчислення - це лише метод надання комп'ютерних ресурсів, а не нова технологія, вони відкрили нові потоки інформації та різновид надання послуг. Спочатку в інформаційних технологіях панували великі обчислювальні машини. З часом ця система поступилася місцем моделі клієнт-серверної архітектури. Сучасна індустрія інформаційних технологій стає все більш мобільною, продуктивною і звичайно більше зосереджено на наданні послуг. Хмарні технології розширюються, і стає все більше хмар постачальників на світовому ринку, таких як Google Cloud, Amazon Cloud та Microsoft Azure. Важливо підкреслити, що сучасна хмара - це платформа, яка пропонує рішення для надзвичайно складного бізнесу. Дане поняття охоплює ряд різних інформаційних систем. Таких, як ERP – це планування ресурсів підприємства, CRM – це управління відносинами з клієнтами, DMS – це система управління документами та багато інших. Все це може представляти собою незалежне або інтегроване рішення.

На сьогоднішній день зображення є невід'ємною частиною нашого життя, адже з кожним роком в мережі інтернет збільшується кількість використаних зображень та відеоматеріалів. Тому питання попередньої обробки і аналізу фотоматеріалів набуває все більш актуального значення. Багато сучасних систем, особливо ті, які зв'язані з медициною почали перехід на підтримку та генерацію зображень. Дані обставини

вимагають від програмного забезпечення витримувати суттєве збільшення обсягу виконуваних робіт без суттєвої втрати швидкодії, можливість роботи, як незалежної системи так і інтеграції в алгоритмічні процеси зовнішніх систем. Спроектowana система повинна забезпечити легкість масштабування та можливість використання сучасних сервісів зокрема хмарних технологій.

З цього випливає наступна мета даної магістерської роботи. Вона полягає в розробці архітектури програмного забезпечення по обробці оцифрованих зображень на основі використання хмарних технологій.

Для реалізації поставленої **мети** сформульовані ряд наступних **завдань**:

- пошук систем, які могли б виступати аналогами та аналіз сучасних рішень даного програмного забезпечення;
- аналіз необхідного функціоналу для роботи з оцифрованими зображеннями, розгляд систем, які б могли інтегрувати розроблене програмне забезпечення;
- вибір стеку технологій, на ринку хмарних технологій, а саме розгляд Google Cloud, Amazon Cloud, Microsoft Azure;
- проектування та реалізація хмарної архітектури по обробці оцифрованих зображень можливістю легкого масштабування;
- аналіз отриманих результатів.

Тема даного магістерської дисертації – це архітектурне рішення для програмного забезпечення обробки оцифрованих зображень за допомогою хмарних технологій.

Об'єктом дослідження виступає процес обробки оцифрованих зображень за допомогою хмарних технологій.

Предметом дослідження є архітектурні рішення та підходи для перетворення та редагування оцифрованого зображення з використанням хмарних технологій.

1 АНАЛІЗ СИСТЕМ-АНАЛОГІВ ПО ОБРОБЦІ ОЦИФРОВАНИХ ЗОБРАЖЕНЬ

1.1 Опис предметної області

В ІТ індустрії можна спостерігати зростання масових і складних систем, які побудовані навколо великого набору даних. Такі системи інтегрують та розробляють додаткові модулі та підсистеми для подальшого аналізу. Багато продуктів почали інтегрувати можливість роботи з зображеннями. Можна спостерігати швидкі зміни, які були спровоковані цією тенденцією, у багатьох сферах. Все це призвело до появи нових сфер та підвищення в фінансовій активності, тобто появи новітніх стартапів, що надають інфраструктуру та інструменти для роботи з ними.

Цифрова обробка зображення стає широкою сферою застосування, яку почали впроваджувати в кожен сферу нашого життя. Однією з найбільших проблем цифрової обробки зображення є те, як зробити обробку фото безпечною та позбутися проблем при збільшенні навантаження на сервер для підтримки додатку у режимі реального часу. Архітектура цифрової обробки зображень потребує високошвидкісних компонентів, які були б спроектовані так, щоб їх можна було легко масштабувати, добавляючи новий функціонал і забезпечити легку можливість інтеграції з іншими сервісами якщо це буде необхідно.

Найважливішою концепцією проектування будь-якої системи є правильний підхід до компенсації: між вартістю і якістю. Тобто якщо говорити про якість зображення, воно має велику кількість пікселів, з іншого боку така велика кількість пікселів потребує високої швидкості обробки, щоб досягти результату. Тому основна проблема більшості цифрових обробок над зображень полягає в поєднанні між програмним і апаратним забезпеченням. Все це допомагає досягти надійності і високої швидкості обробки. В свою чергу збільшення розміру зображення може призвести до збільшення кількості пікселів на одне зображення означає. З цього

впливає, що необхідно більше часу на обробку, але, з іншого боку, це дає хорошу якість.

Корпорації готові працювати та інтегрувати свої продукти з системами по роботі з фото, адже візуалізація продукту дає можливість більш ефективно продавати та отримувати прибуток.

Також одним з аспектів який би хотілось відмітити, що певні зображення є чорно-білими та й в цілому, існує можливість надати покращення та підвищення якості цього зображення. В чорно-білих оцифрованих зображеннях можна зустріти різноманітні дефекти. Навіть під час оцифрування може статись якісь пошкодження. Такі проблеми звийно ж впливають при перетворенні чорно-білого зображення в кольорове. Для ефективного вирішення даної проблематики використовують такі поняття як цифрова реставрація, яка на базі нейронних мереж вирішує дану проблематику.

1.2 Аналіз існуючих рішень

В ході аналізу і пошуку існуючих рішень були знайдені окремі сервіси за допомогою яких можна здійснювати обробку зображення, а також проводити кольоризацію фото, але на просторах інтернету не зустрічається сервісу, який підтримував можливість інтеграції з іншими сервісами, а також можливість редагування та перетворення зображення.

1.2.1 Сервіс Colorize B&W Photos

Сервіс Colorize B&W Photos [1] – це невеликий сайт, який був створений компанією Photomyne Ltd, інтерфейс даного сервісу зображений на рис. 1.1.

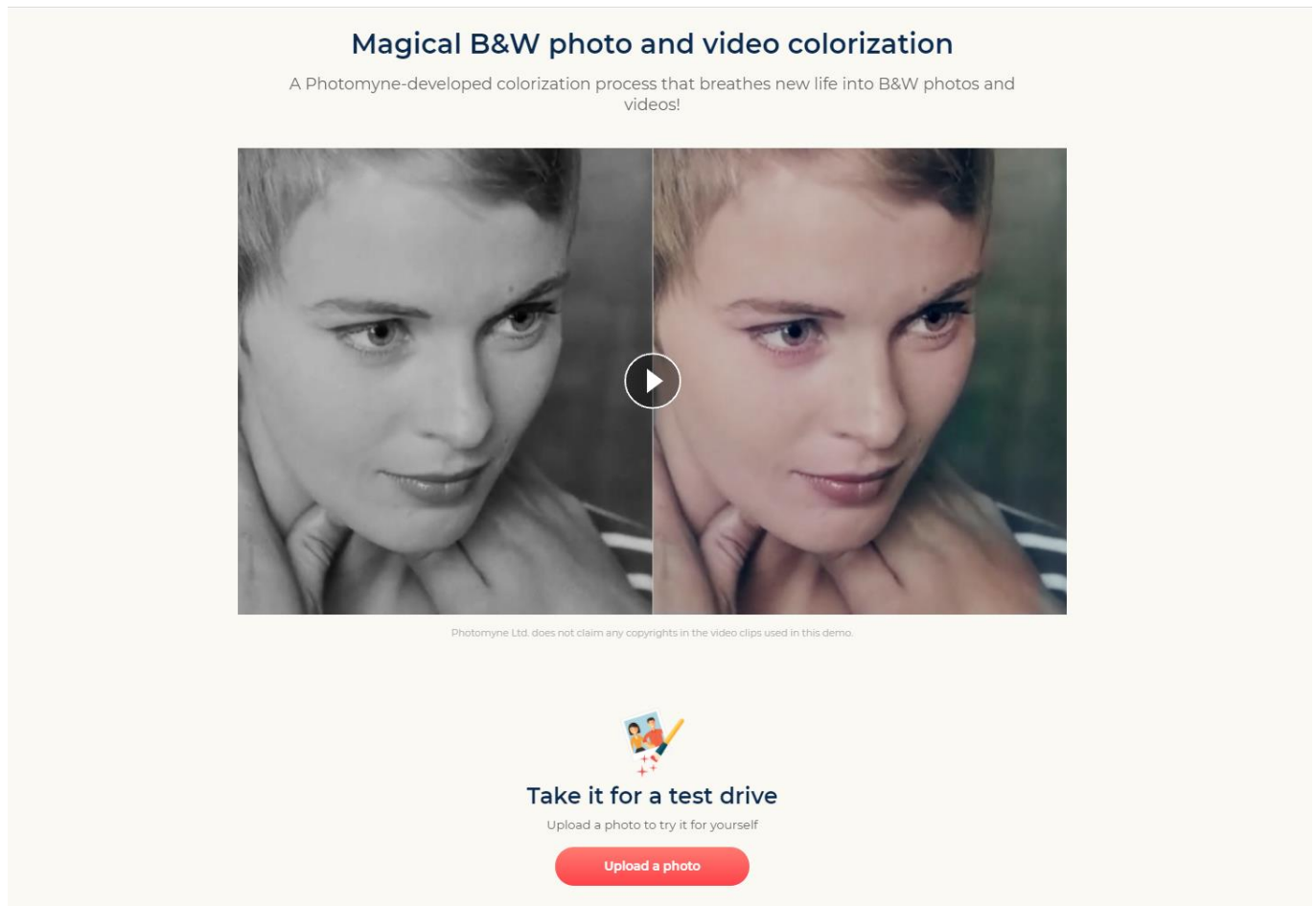


Рисунок 1.1 — Головний інтерфейс сервісу Colorize B&W Photos

Даний сервіс використовує неймережу для перетворення чорно-білих зображень в кольорові, а також є можливість перетворення невеликих чорно-білих відео.

Якщо говорити за набір можливостей Colorize B&W Photos, то можна виділити наступні переваги, а саме:

- простий та зрозумілий у використанні інтерфейс;
- швидкість роботи алгоритму;
- нейронна мережа навчається в процесі обробки.

Для наглядного прикладу та перевірки роботи сервісу було знайдено оцифроване чорно-біле зображення і на її основі зроблений тест. Зображення оригінальне на рис. 1.2.



Рисунок 1.2 — Чорно-біле зображення, яке використовують для перевірки сервісу

Після очікування маємо наступні результати, які зображені на рис.1.3. Якщо оцінювати критерії, після перетворення, то для даного зображення, можна виділити наступні, які явно проявились, а саме дане зображення має природні кольори, історичну достовірність та реалістичність.



Рисунок 1.3 — Результат після кольоризації сервісом Colorize B&W Photos

Окрім переваг також є і недоліки даного сервісу, адже відразу відчутні недоліки після перетворення даного зображення. Результати будуть трішки кращими, якщо дане зображення ще раз перевикористати в сервісі. Даний сервіс має ліміт в кількості завантажень має платний контент та додає водяні помітки компанії до зображення та не приймає картинки розміром більше ніж 1 мб. Відсутня можливість якось редагувати фото, тобто збільшувати, зменшувати, робити повороти чи обрізку даного зображення. А також не можливо інтегрувати даний сервіс з іншими продуктами.

1.2.2 Сервіс Pho.to

Даний редактор можна використовувати після того як було обрано мову користування, головний інтерфейс зображений на рис.1.4.

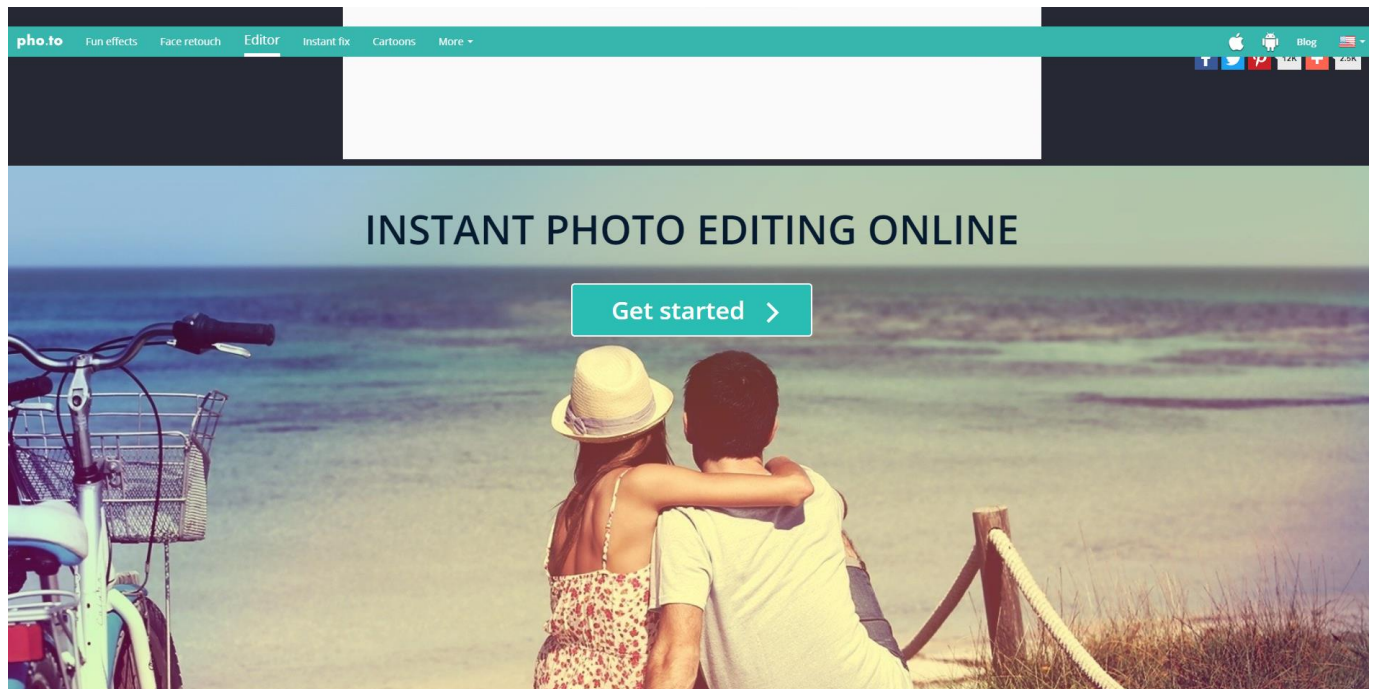


Рисунок 1.4 — Головний інтерфейс сервісу Pho.to [2]

Після завантаження зображення цей додаток дає зрозуміти, що він охоплює багато онлайн-інструментів для редагування зображень.

Даний сервіс вирішує проблему оновлення зображення, зміни розміру, редагування чи додавання тексту до фотографій, а також надає можливість легко створювати фотоколажі. Перелічений функціонал зображений на рис. 1.5.

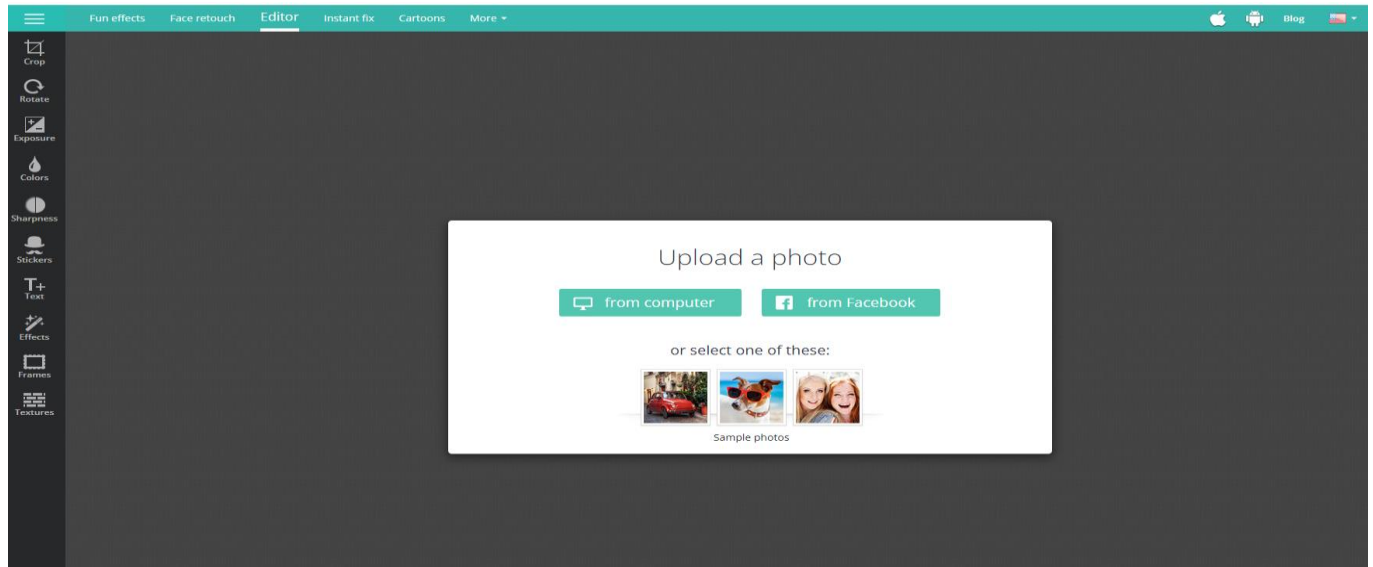


Рисунок 1.5 — Головний інтерфейс сервісу Pho.to [2]

Недоліком даного сервісу є відсутність функції подальшого зберігання зображення у власному обліковому записі. Тільки після редагування існує можливість збереження даного зображення на свою робочу станцію. Також відсутня можливість інтеграції з іншими продуктами та можливість кольоризації зображення.

1.2.3 Сервісу PhotoshopOnline

PhotoshopOnline [3] має наступний графічний інтерфейс головного екрану який зображений на рис.1.6.

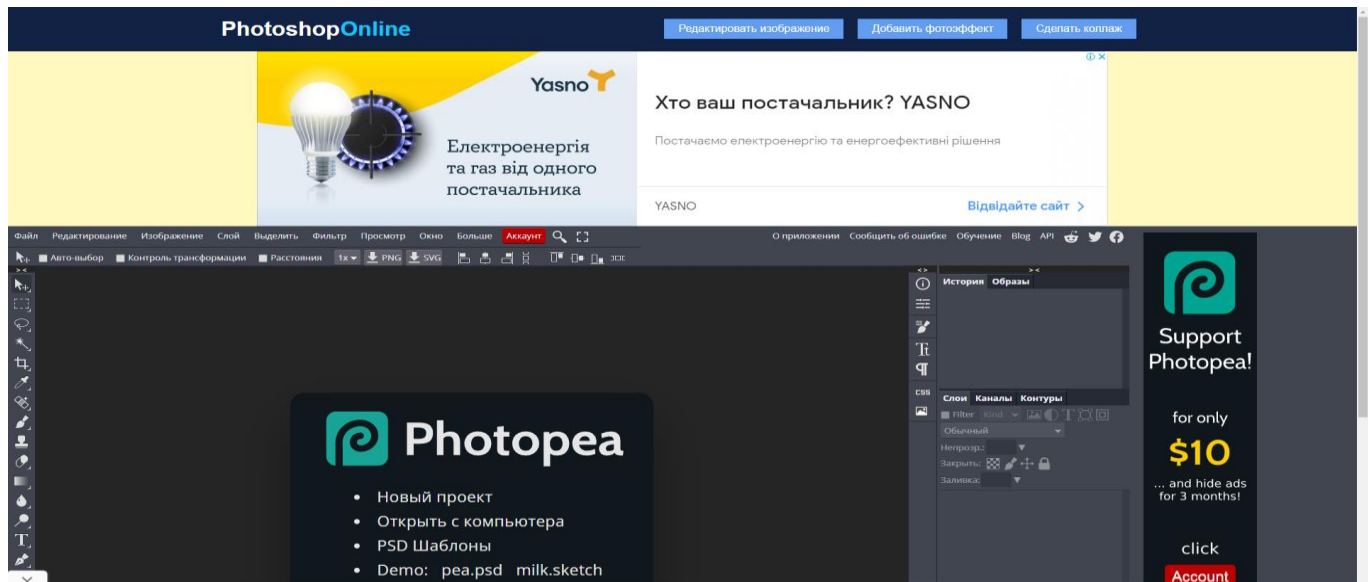


Рисунок 1.6 — Головний інтерфейс сервісу PhotoshopOnline [3]

Цей фоторедактор дозволяє обробляти зображення онлайн безкоштовно, навіть не потрібно встановлювати програму на робочий пристрій. Недоліками даного сервісу слід вважати відсутність локалізації.

У графічному редакторі онлайн можна швидко та з легкістю завантажувати та редагувати ваші зображення, відео. Якщо говорити за інструменти, то наявний весь базовий набір графічних інструментів і функцій для подібних програм. Сервіс надає можливість підтримки готових шаблонів для різноманітних колажів, створення плакатів, різних рекламних банерів, та багато чого іншого. І все це з легкістю можна буде відправити на друк у високому якості.

Даний фоторедактор окрім можливостей редагування зображень надає також корисні статті та навчальний матеріал як правильно використовувати той чи інший функціонал з підказками як зробити зображення кращим приклад таких можливостей продемонстровано на рис.1.7.

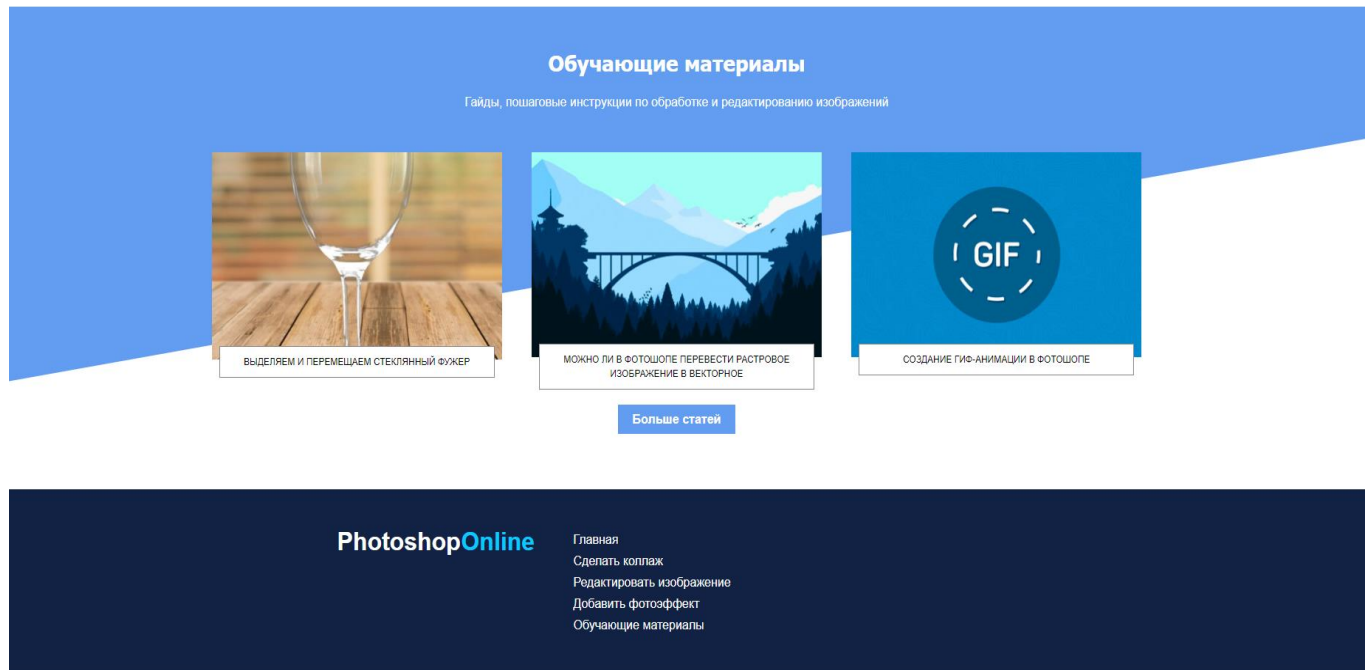


Рисунок 1.7 — Приклад по навчальних матеріалах на сайті

Даний сервіс має приємний користувацький інтерфейс, але він також якщо розглядати функціональні можливості окрім редагування зображень, не може ніяк кольоризувати зображення та надавати можливість інтеграції з іншими продуктами.

1.2.4 Сервіс «Комп'ютерний зір»

Компанія Mail.ru має різноманітні продукти по комп'ютерного зору. Одним з таких продуктів є продукт Vision. Окрім того, Mail займаються розпізнаванням особистих даних з камер відеоспостереження. Vision є інтегрованою технологією для розпізнавання фотографій у додатку mail ru. Крім цих можливостей і більше того, пропонує обробку чорно-білих фото та перетворення її в кольорове зображення. Для перевірки роботоспроможності даного сервісу необхідно використання стороннього VPN застосунку. На рис.1.8 наведено приклад перетворення зображення з чорно-білого в кольорове з використанням продукту Vision.



Рисунок 1.8 — Результати роботи сервісу Комп'ютерний зір від Mail.ru

Відразу можна виділити, що великий мінус даного сервісу є його швидкість. Адже використання складних алгоритмів для покращення якості й відновлення від пошкоджень – все це призводить до відчутної затримки при роботі.

Таблиця 1.1 – Порівняння функціональних можливостей сервісів по роботі з фото

Параметри оцінювання	Colorize B&W Photos	Комп'ютерний зір від Mail.ru.	PhotoshopOnline	Pho.to
Колоризація зображень	+	+	-	-
Вільний доступ без використання сторонніх засобів	+	-	+	+
Обрізка зображення	-	-	+	+

Продовження таблиці 1.1

Базовий набір графічних інструментів	-	-	+	+
Використання хмарного архітектурного рішення	-	-	-	-
Наявність особистого кабінету	-	+	-	-
Відсутність токенів та бази користувачів	-	+	-	-
Можливість перегляду попередніх зображень	-	-	-	-
Можливість інтеграції системи з іншими продуктами	-	-	-	-

Було прийнято рішення об'єднати весь необхідний функціонал, який є в аналогів.

1.2.5 Переваги використання хмарних технологій

Вплив хмарних технологій на бізнес є величезний з точки зору надання можливості як малим так і великим компаніям. Відразу відкривається можливість легко використовувати інноваційні технології у своїй бізнес-моделі. Зникає потреба в експертах різних сфери, які були б необхідні, якщо у компанії була своя інфраструктура. Переваги хмарних технологій відразу відчутні в таких сферах як:

технічній та фінансовій. Головною перевагою у довгостроковій перспективі є інвестиційний сегмент та вартісний. Адже сплата працює за принципом оплати по факту використання.

На додаток до цього, є багато інших переваг, які пов'язаних зі зменшенням часу розгортання, витрат на штаб співробітників, а потім повне технічне забезпечення, витрати на обслуговування тощо. Однією з найважливіших переваг даних рішень є гнучкість. Крім того, компанії можуть використовувати новітні технології без великих інвестицій, тобто немає бар'єру для заміни і переходу на нові технології навіть для малих компаній.

Розглянемо основні переваги хмарних технологій:

– різке зниження витрат, відразу після застосування хмарних технологій у вашому проекті знижується рівень витрат через контроль шляхом моніторингу використаних ресурсів;

– pay-as-you-grow, дана хмарна технологія надає можливість збільшити потужність та ресурси відповідно до вимог компанії, тобто, якщо великі інвестиції в інфраструктуру на даний момент часу не потрібні і не є критичними для компанії, то за них платити не потрібно, а в подальшому при розширенні можна з легкістю це зробити, якщо це знадобиться в майбутньому;

– поняття еластичність – це можливість масштабувати обчислювальну потужність залежно від потреби, завдяки цій можливості, компанія оплачує корисні ресурси залежно від потреб і вимоги;

– гнучкість – використовуючи хмарні технології, компанії розширюють бізнес гнучкість завдяки відсутності вузьких місць, з іншої сторони, вони мають можливість без значних випробувань та розробки нових послуг інвестиції;

– обслуговування та витрати на внутрішню інфраструктуру – цей пункт вартий уваги, адже володіння власними інфраструктура створює більше витрат, при переході на використання хмарних технологій багато зобов'язань та різноманітних ризиків

лягає на постачальника хмарних послуг, все права та зобов'язання по обслуговуванні визначені в угодах про рівень обслуговування;

- є можливість самообслуговування за вимогою – тобто, можливість забезпечує обслуговування інфраструктури на вимогу, вона надає можливість використання ресурсів масштабно, де компаніям не доведеться турбуватися про обмеженість простору чи потужність процесора тощо;

- широкий доступ до мережі – в цілому доступ до хмарної технології можливий в будь-який час за умови, що користувач має відповідний доступ до мережі інтернет;

- об'єднання ресурсів – постачальники хмарних технологій мають великі потужності обчислювальних ресурсів, найчастіше ці ресурси розподілені між кількома центри обробки даних у різних місцях [4].

- можливість виміряти сервіс – всі ресурси, які будуть експлуатуються, можуть бути вимірні дуже легко, тобто користувач може оптимізувати свої витрати, а також ресурси, необхідні для послуг, які він використовує;

- надається автоматичне оновлення програмного забезпечення, існує проблема в оновленні апаратного та програмного рівня, застосування хмарних технологій усуває ці місця.

1.3 Висновки до розділу 1

У даному розділі сформовано опис предметної області та проведено аналіз існуючих рішень, які пропонує ринок і на основі цих рішень було проаналізовано і сформовано основні вимоги для подібних систем. Також було проаналізовано та наведено переваги використання хмарних технологій для побудови програмного забезпечення. При аналізі існуючих аналогів було визначено переваги та недоліки кожної системи. Виявлено потребу в проектуванні системи на основі хмарних технологій, яку можна було б інтегрувати з іншими продуктами, надавши можливість підвищення горизонтального масштабування та зменшення ресурсів для підтримки працездатності серверу.

2 ПОРІВНЯННЯ ХМАРНИХ СЕРВІСІВ ТА ОГЛЯД ТЕХНОЛОГІЙ ДЛЯ ПОБУДОВИ АРХІТЕКТУРИ

2.1 Розгляд провідних хмарних компаній

На сьогоднішній день відбувається жорстка конкуренція за лідерство у публічних хмарних обчисленнях, яка затронула такі корпорації: веб-служби Amazon проти Microsoft Azure та проти Google Cloud Platform, логотипи даних компаній зображено на рис. 2.1.



Рисунок 2.1 — Логотипи провідних компаній в хмарних обчисленнях [4]

Очевидно, що ці три провідні хмарні компанії займають лідируючі позиції на ринках інфраструктури як послуги або IaaS та платформи як послуги або PaaS [5].

AWS є особливо домінуючим, адже згідно зі звітом Synergy Research Group за 2021 рік, Amazon продовжував відображати загальне зростання ринку, тому він зберіг

свою частку в 35% світового хмарного ринку [10]. Друге місце в рейтингу Microsoft знову зростає швидше, ніж ринок, і його частка на ринку зростає майже на три відсотки за останні чотири квартали, досягнувши 20%. Тим часом Microsoft особливо сильна в SaaS, тоді як Google Cloud, з його потужністю у сфері штучного інтелекту, позиціонується для агресивного зростання в міру зростання ринку штучного інтелекту – і відомий тим, що пропонує знижки [9].

Розглянемо порівняння хмари між AWS та Azure та Google. Якщо говорити за AWS завдяки величезному набору інструментів, який продовжує зростати в геометричній прогресії, можливості Amazon не мають собі рівних. Проте структура витрат може бути заплутаною, а його зосередженість на загальнодоступній хмарі, а не на гібридній або приватній хмарі означає, що взаємодія з вашим центром обробки даних не є головним пріоритетом AWS.

В свою чергу Azure є близьким конкурентом AWS з більш здатною хмарною інфраструктурою. Якщо ви корпоративний клієнт, Azure у випадку проблеми зможе забезпечити зворотній зв'язок для її вирішення вашою мовою. Небагато компаній мають корпоративний досвід і підтримку Windows. Azure, керує центром обробки даних, і платформа Azure старанно працює над інтеграціями з центрами обробки даних Windows.

Аутсайдером у цій конкурентній боротьбі є Google. Він пізніше за всіх вийшов на ринок хмар і не має такої великої концентрації на підприємстві, яка допомагає залучити корпоративних клієнтів [7]. Але його технічний досвід є достатньо глибоким, а його провідні інструменти в галузі глибокого навчання та штучного інтелекту, машинного навчання та аналітики даних будуть нести великою перевагою.

2.2 Переваги та недоліки AWS, Azure та Google

Кожен з провідних постачальників має певні сильні та слабкі сторони. Спираючись на кожну з цих сторін, можна зробити хороший вибір, який буде

показувати хороші результати для унікальних проєктів. Адже не існує універсального хмарного рішення.

2.2.1 Переваги та недоліки AWS

Найбільшою та головною перевагою Amazon є її домінування на ринку публічних хмар. Провідна консалтингова компанія Gartner назвала AWS лідером на ринку хмарних IaaS в 2021 році. Частково причиною його популярності, безсумнівно, є величезний розмах його діяльності [19]. AWS має величезний і зростаючий набір доступних послуг, а також найширшу мережу центрів обробки даних у всьому світі. У звіті Gartner підсумовано, що AWS — це найбільш розвинутий та готовий до підприємства постачальник, з найглибшими можливостями для керування великою кількістю користувачів і ресурсів [11]. Великі недоліки Amazon стосуються вартості. Хоча AWS регулярно знижує ціни, багатьом підприємствам важко зрозуміти структуру витрат компанії та ефективно керувати цими витратами під час виконання великого обсягу робочого навантаження на сервіс. Однак загалом ці недоліки більш ніж переважають сильні сторони Amazon, і організації будь-якого розміру продовжують використовувати AWS для широкого спектру робочих навантажень.

2.2.2 Переваги та недоліки Microsoft Azure

Пізніше після AWS вийшла Microsoft на ринок хмар. Вона дала собі швидкий старт, адже вона взяла своє локальне програмне забезпечення. Тобто за основу було взято наступне забезпечення: Windows Server, SQL Server, Dynamics Active Directory, .Net та багато інших і все це було перепрофільовано для хмари. Головна причина успіху Azure через те що багато підприємств працюють з Windows та іншим програмним забезпеченням Microsoft. Оскільки Azure тісно інтегрована з цими іншими програмами, то підприємства, які використовують багато програмного забезпечення Microsoft, дають перевагу використанню Azure в порівнянні з іншими аналогами. Адже тільки для них має сенс використовувати Azure. Це зміцнює

лояльність наявних клієнтів Microsoft. Крім того, якщо ви вже є корпоративним клієнтом Microsoft, очікуйте значних знижок на контракти на обслуговування. З іншого боку, Gartner дорікає до деяких недоліків платформи [12]. Хоча Microsoft Azure є корпоративною платформою, клієнти Gartner повідомляють, що досвід роботи з обслуговуванням виглядає менш підготовленим для підприємства, ніж вони очікували. Клієнти посилаються на проблеми з технічною підтримкою, документацією, навчанням та широтою партнерської екосистеми ISV.

2.2.3 Переваги та недоліки Google Cloud Platform

Якщо говорити про Google, то він має потужну пропозицію щодо контейнерів, оскільки Google розробила стандарт Kubernetes. Цей стандарт зараз пропонують AWS та Azure. GCP спеціалізується на високоякісних обчислювальних пропозиціях, таких як великі дані, аналітика та машинне навчання [14]. Він також пропонує значний баланс навантаження та балансування навантаження - Google знає центри обробки даних та швидкий час відгуку. З іншого боку, Google займає третє місце за часткою ринку, можливо, тому, що у нього немає традиційних відносин з корпоративними клієнтами. Однак він швидко розширює як свої пропозиції, так і свої глобальні центри обробки даних. Gartner сказав, що його клієнти зазвичай обирають GCP як вторинного постачальника, а не стратегічного постачальника, хоча GCP все частіше обирається як стратегічна альтернатива AWS клієнтами, чий бізнес конкурує з Amazon, і які більше орієнтовані на відкриті вихідні коди або DevOps-орієнтовані і отже, менш добре пристосовані до Microsoft Azure.

Підведемо загальні підсумки в таблиці 2.1.

Таблиця 2.1 – Порівняння переваг та недоліків хмарних корпорацій

Хмарні корпорації	Переваги	Недоліки
AWS	<ul style="list-style-type: none"> – домінуюча позиція на ринку; – широкі та зрілі пропозиції; – підтримка великих корпорацій; – широке навчання; – глобальне охоплення; 	<ul style="list-style-type: none"> – важко використовувати; – вправління витратами; – переважні варіанти;
Microsoft Azure	<ul style="list-style-type: none"> – другий за величиною постачальник; – інтеграція з інструментами та програмним забезпеченням Microsoft; – широкий набір функцій; – гібридна хмара; – підтримка відкритого коду; 	<ul style="list-style-type: none"> – проблеми з документацією; – неповні інструменти управління;
Google Cloud Platform	<ul style="list-style-type: none"> – розроблено для компаній, що працюють у хмарі; – прихильність відкритому коду та портативності; – великі знижки та гнучкі контракти; – досвід DevOps; 	<ul style="list-style-type: none"> – пізній вихід на ринок IaaS; – менше функцій та послуг; – історично не було орієнтовано на підприємство;

Також доречним порівнянням буде ціна за запитом. Коли справа доходить до ціноутворення, у кожній з платформ є дві спільні речі [13]. Перше - це безкоштовний

рівень із суворо обмеженими опціями та погодинна або хвилинна структура ціноутворення на вимогу для всіх ресурсів. На жаль, ціна може сильно відрізнятись. Все це залежить від використання ресурсів, вибору послуг. Ось чому порівняння та вибір може бути складним. Якщо говорити про узагальнення функції безкоштовного рівня та порівняти структури цін кожної служби, отримуємо наступне.

Ціни в AWS та структура ціноутворення, є доволі складними, що вам знадобиться сторонній додаток для керування нею. В свою чергу Amazon пропонує 12-місячний термін служби EC2 по 750 годин на місяць з безкоштовним рівнем і знижкою до 75% за зобов'язання підпису контракту на 1-3 роки [14].

Ціни Azure є подібними до AWS, користувачі Azure часто також використовують сторонні додатки для підрахунку управління витратами. Azure пропонує 12 місяців використання віртуальних машин з 745 годин на місяць із своїм безкоштовним рівнем. Також наявна знижка за зобов'язання використання на 1–3 роки.

Відразу можна сказати, що Google зрозумів всю тенденцію і намагався вчитися на помилках своїх конкурентів. Першим кроком це було впровадження порівняно просту модель витрат. Крім того, GCP включає в себе один безкоштовний мікро екземпляр на місяць. Вже після першого року використання надає знижку 30% за тривале використання [15].

Вибір компанії постачальника перш за все диктуються правилом вибору більшості. Адже ментальний підхід, до того що багато людей використовують їх продукт, людина починає вже довірливо ставитись до продукту. А після аналізу цін і відсутності спецефічної потреби у використанні GCP чи Azure починає використовувати AWS.

2.3 Математичне забезпечення

2.3.1 Поняття шарів згорткової мережі

Згорткова мережа складається з невеликих будівельних блоків, так званих шарів. Кожен шар, в свою чергу, діє як єдиний крок у загальній трансформації вихідного результату. Ці шари можуть приймати різні форми і можуть служити для багатьох цілей. Традиційно в згорткових мережах шар приймає вхідні дані, так як масив, що представляє зображення і видає результат, реалізуючи функцію вигляду:

$$y = \delta(Wx + b)$$

де $x \in \mathbb{R}^n$, а $y \in \mathbb{R}^m$ – це входи та виходи шару відповідно, в свою чергу $W \in \mathbb{R}^{n \times m}$ – ваги, пов'язані з шаром, $b \in \mathbb{R}^m$ – вектор зміщення, який виконує завдання адитивного компонента перетворення, та $\delta : \mathbb{R} \rightarrow \mathbb{R}$ є нелінійним входом функція активації.

Шар може приймати більше ніж один вхід і створювати більше ніж один вихід, котрий можна представити як конкатенацію входів або виходів для отримання єдиного входу або виходу. Вага матриці та вектор зміщення також називаються параметрами шару. Вони отримані шляхом зворотнього поширення втрат, що обчислюються як похибка між прогнозуванням мережі та правдивими навчальними даними [36].

Функція активації надає мережі можливість вивчати нелінійні відображення, оскільки компоненти ваги та зміщення здатні створювати лише лінійні перетворення. Найпоширеніші приклади функцій активації включають сигмоїдну функцію:

$$\delta(x) = \frac{1}{1+e^{-x}}$$

гіперболічна функція, така як гіперболічний тангенс:

$$\delta(x) = \tanh(x)$$

Особливо у випадку з згортковими мережами, функція ReLU обчислюється ось так:

$$\delta(x) = \max(0, x)$$

Кожна з цих функцій призведе до впровадження нелінійності в мережу. Також можна представити функцію активації як окремий шар з ідентичною матрицею W і нульовим вектором b - це звичайна практика в багатьох згорткових мережах.

Згорткові шари є основними шарами, які використовуються в згорткових мережах. Вони представляють операцію згортки, що виконується на вході з вагами як параметри згорткового ядра, у двовимірному випадку:

$$y_{i,j} = \sum_{m=i-\frac{k}{2}}^{i+\frac{k}{2}} \sum_{n=j-\frac{k}{2}}^{j+\frac{k}{2}} y_{m,n} W_{m=i-\frac{k}{2}, n=j-\frac{k}{2}} + b_{m=i-\frac{k}{2}, n=j-\frac{k}{2}}$$

де k - це розмір ядра згортки також називається фільтром, а W і b просто виконуються від першого індексу до останнього. Як ми бачимо, ваги зберігаються для ядра, а не для входу, тобто шару не потрібно мати вхідний розмір фіксованого розміру, який діє як алгоритм sliding-window. Вагові коефіцієнти просторово ідентичні для всіх частин вхідних даних, що робить зсувні шари інваріантними.

Ці два показники також називають шириною і висотою. Якщо відштовхуватись від даного, рис. 2.2, основним гіпер параметром згортки є розмір самого ядра. Зазвичай даний розмір позначається як число або описується як множення, тобто 4×4 для ядра розміру 4, як приклад.

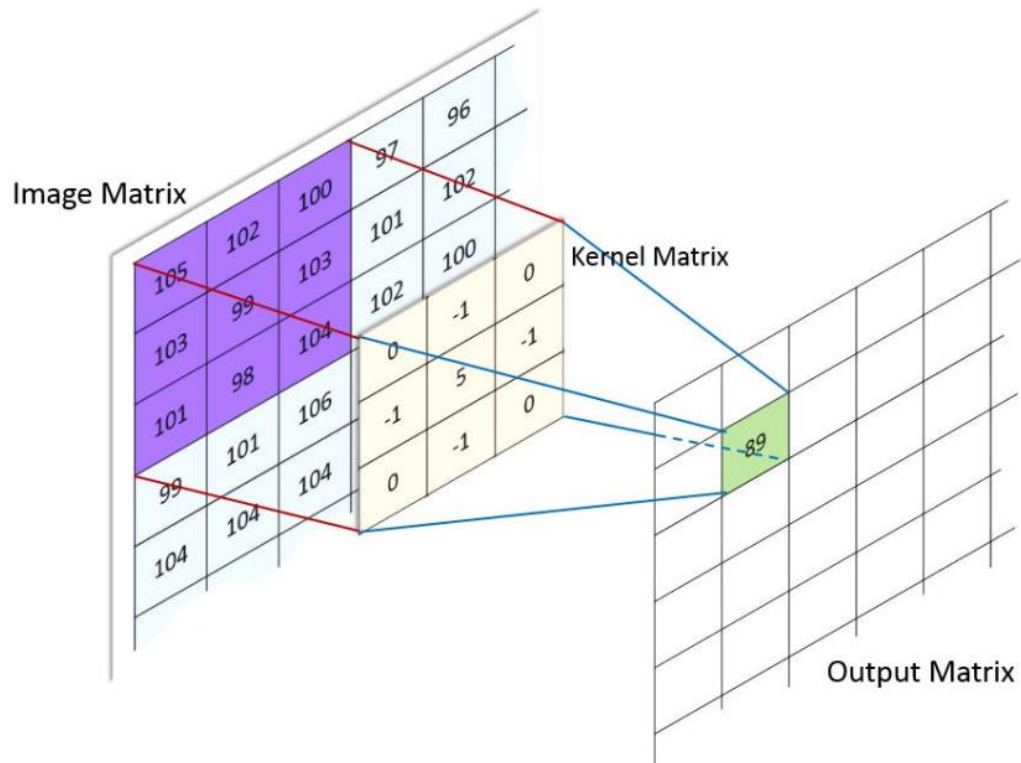


Рисунок 2.2 — Схематичне зображення згортки [8]

Що стосується результату, існує кілька гіпер параметрів, традиційно пов'язаних з шаром згортки. Перший параметр - це шар i як правило, він не завжди складається з одного ядра, але замість цього навчаються декілька фільтрів ядра, а їхні виходи з'єднані у тривимірний об'єм. Цей параметр називається кількістю виходів. Якщо під'єднати до іншого згорткового шару, то останній шар розглядає об'єднаний вимір як вимір фіксованого розміру. Спочатку вивчає всі його фільтри по довжині глибини.

Ідея розрахунку згортки полягає в тому, що не потрібно виконувати для кожної точки введення. Тобто, допускається пропустити кожну n точку входу в будь-якому напрямку. Цей підхід називається *stride*. В цьому прикладі буде використовуватись вихід із кроком $n + 1$. *Stride* є важливою концепцією. Вона, дозволяє нам зменшити вибірку вхідних даних швидко для вирішення, шляхом згортання. Збільшення дискретизації, може бути реалізовано шляхом транспонованої згортки, де вхідна

матриця розповсюджується шляхом нульового заповнення. Що допомагає створювати вихідну роздільну здатність вище, ніж вхідна роздільна здатність.

Також можна помітити, що згортки не можна обчислити по краях вхідних даних, оскільки відсутні значення для обчислення. В результаті вихід буде $\frac{k}{2}$, тобто він буде менший уздовж кожного краю. Якщо враховувати ввід 256×256 , виконання нейронної мережі із розміром ядра 7 призведе до розміру виводу 250×250 . Щоб запобігти цьому, до входу додається padding, таким чином існує можливість обчислити вихід навіть для крайових точок. Padding може бути просто заповнений нулем. Сформуємо остаточну формулу підрахунку виводу:

$$w_o = \frac{w_i - k + 2p}{s} + 1$$

В даній формулі w_o - це ширина виводу, w_i - ширина входу, k - розмір фільтра ядра, p - це розмір відступу краю, s - це крок. Нейронна мережа, в якій використовується багато згорткових шарів, називається глибинною згортковою нейронною мережею або CNN [37].

Елементні шари - це шари для основних операцій. Вони беруть n входів однакової форми та виконують функцію по елементам. Ці функції найчастіше є множенням або додаванням, але можуть також включати функцію \max або \min . Вихід для двовимірного випадку розраховується як:

$$y_{i,y} = f(x_{1,i,j}, \dots, x_{10,i,j}, \dots, x_{n,i,j})$$

в даному випадку f - це виконана функція.

2.3.2 Тренування моделі

Разом параметри всіх шарів нейронної мережі називають параметрами мережі або ваговими коефіцієнтами нейронної мережі. Дані параметри потрібно вивчити з даних, які надає користувач. Ну й сам процес набору ваг - називається навчанням. Коригування ваги в згорткових нейронних мережах буде здійснюватись за допомогою процесу зворотного поширення. Backpropagation можна узагальнити і поділити на такі підпункти: прямий прохід, функція втрати, зворотний прохід та оновлення ваги. Під час прямого проходження береться навчальний приклад x і він проходить через шари нейронної мережі, щоб отримати певне передбачення. На основі цього прогнозу буде обчислено похибку між передбаченням і істиною навчального прикладу, виміряну як значення обраної функції втрат.

Оскільки завжди стоїть ціль мінімізувати втрати, необхідно оновити ваги у всіх шарах мережі, тому й виконується зворотний прохід. Для цього необхідно обчислити дискретні часткові похідні між вагами нейронної мережі, що виводить функцію втрат як $\frac{dLoss}{dw} = f_v$ і на етапі оновлення ваги оновлюємо вагу таким чином:

$$w = w_i - \eta f_v$$

де η — це гіпер параметр швидкості навчання, а w_i — поточні ваги. Ця концепція застосовується до кожного шару в зворотному напрямку прямого проходу, замінюючи $dLoss$ для часткових похідних попереднього шару на кожному кроці. На практиці робота з даною формулою буде складнішою, адже вона поступово вдосконалювалося в міру того, як було проведено більше досліджень. Також на це буде впливати гіпер параметри алгоритму оптимізації [38].

Функція втрат нейронної мережі буде визначатись як функція її прогнозу та основної істини. Значення цієї функції буде вимірювати похибку передбачення по відношенню до попередньо визначеної мітки істини. Цю функцію можна

налаштувати відповідно до завдань, які повинна засвоїти мережа, але на практиці існує кілька часто використовуваних функцій, таких як Евклідова норма L_2 або втрата Крос-ентропії.

Функція втрат повинна відповідати двом припущенням, щоб її можна було застосувати до будь-якого алгоритму навчання. Перше, вона повинна мати можливість розкласти втрату всього навчального набору на середнє значення функцій втрат окремих прикладів навчання:

$$F = \frac{1}{n} \sum_x F_x$$

Дане припущення необхідно для того, щоб мати можливість обчислити часткові похідні функції втрат для окремого навчального прикладу. Дана ітеративна оптимізація даних має назву стохастичним градієнтним спуском або SGD. Вона призведе до використання в алгоритмі зворотного розповсюдження [40].

Друге припущення полягає в тому, що це повинно бути функцією вихідного сигналу нейронної мережі. Слід зауважити, що повторення над усіма прикладами занадто багато разів призводить до перетренування. Тобто ваги навчаються кодувати навчальний набір даних, а не добре узагальнювати і аналізувати на небачених прикладах, особливо якщо набір даних невеликий [6].

Також потрібно сказати пару слів про нормалізацію. Під час навчання згорткової нейронної мережі існує два типи нормалізації, які зазвичай виконуються. Перший тип стосується нормалізації вхідних даних. Потрібно переконатись, що будь-який вхідний сигнал, що надходить у мережу, стискається або змінюється на ті самі діапазони. Це необхідно для глобального гіпер параметру швидкості навчання. У випадку, якщо деякі вхідні дані працюють на іншій шкалі значень, тобто значно важче тренувати дані рівномірно. Крім того, корисно масштабувати всі вхідні дані в

діапазоні від 0 до 1. Є декілька причин чому так. Адже нормалізація призводить до поліпшення швидкості конвергенції [39]. Поліпшення відбувається через те, що абсолютний розмір вхідних даних доведеться коригувати шляхом налаштування швидкості навчання мережі, а більш високі значення швидкості навчання можуть призвести до коливання ваги. Інша причина вона є суто технічною. Якщо врахувати, що ваги зазвичай представлені як числа з плаваючою комою подвійної точності, використання денормалізованого інтервалу може підвищити точність, оскільки дозволяє більш точно вивчати параметри.

Другий тип нормалізації стосується внутрішнього коваріантного зсуву. Під час навчання нейронної мережі розподіл абсолютних значень кожного входу мережі зміниться, адже, зміняться параметри попередніх рівнів. Це уповільнює навчання, вимагаючи частих змін швидкості навчання, а ефект особливо яскраво проявляється при використанні функції активації, а саме, ReLU. Також можливо зробити крок пакетної нормалізації частиною самої архітектури і виконувати нормалізацію на основі кожного рівня або між кожним блоком шарів. Нормалізація зазвичай проводиться шляхом віднімання очікуваного значення та ділення на стандартне відхилення. Ці значення розраховуються та зберігаються як статистичні дані для кожної групи навчання.

2.3.3 Ініціалізація моделі

Для того, щоб почати навчання мережі, необхідні початкові ваги, як відправна точка. Ініціалізація вагів сильно впливає на швидкість конвергенції, а неправильна ініціалізація може призвести до неправильної роботи градієнтів. Загальні ініціалізація включає в себе ініціалізацію з нульовою вагою. Але це виявляється згубним для швидкості конвергенції, навіть в деяких випадках все це може призвести до того, що мережа ніколи не зблизиться. Адже виконання прямого передавання може заперечити будь-який вплив вхідних даних на виходи. На практиці хороша ініціалізація - це

використання випадкових ваг, нормованих для розміру вхідних та вихідних даних. Коректно використовувати підхід Ксав'є:

$$W = rand(N\left(0, \frac{2}{n_{in} + n_{out}}\right))$$

В даній формулі n - це розмір вхідних і вихідних даних шару. Функція `rand` використовується для генерації випадкової вибірки з нормального розподілу. Ця ініціалізація виробляє числа малого абсолютного значення, як позитивного, так і негативного. Все це вводить асиметрію в роботу, якої не вистачає ініціалізації з нульовою вагою [22].

Останнім часом відбувся розвиток у сфері навчання, залежного від даних ініціалізація, заснований на алгоритмах навчання без нагляду [8]. Також було показано, що цей підхід значно покращує швидкість збіжності та створює ініціалізацію, при якій більшість одиниць у нейронній мережі навчаються приблизно з однаковою швидкістю, це допомагає уникнути проблем з градієнтом. В цілому він запобігає величезним відхиленням у швидкості навчання за параметром під час оптимізації.

2.3.4 Оптимізація виконуваного алгоритму

Алгоритм оптимізації використовується для фактичного оновлення ваг відповідно до формули. Оновлення ваг без будь-яких змін у формулі призводить до звичайного алгоритму SGD, який, як було помічено, має кілька проблем. Для боротьби з ними використовуються різні концепції. Одне з таких рішень – імпульс. SGD може бути дуже повільним при знаходженні локальних мінімумів. Додавання моменту імпульсу допомагає оптимізувати SGD приблизно в тому ж напрямку і запобігає коливанням. При додаванні моменту імпульсу формула змінюється на:

$$w = w_i - u_i$$

$$u_i = \gamma u_{i-1} + \eta f_v$$

де γ – це коефіцієнт імпульсу, а u_i – це поточний вектор швидкості, який має той самий розмір, що й оновлені вагові коефіцієнти. Тобто для кожної ваги існує окремий параметр, що дозволяє SGD оновлювати окремі ваги. Для цього він використовує кроки різної величини. Було розроблено безліч інших алгоритмів, які вирішують різні проблеми, від яких страждає SGD, часто будуючи ці рішення на основі попередніх удосконалень [6].

Найпопулярнішим алгоритмом, який використовують для оновлення ваг при навчанні згорткової мережі є Adam або його повна назва Adaptive Moment Estimation. Adam обчислює адаптивну швидкість навчання для кожного параметра окремо. Окрім того він ефективно розширює ідею глобальної швидкості навчання до ваг і значно зменшує переваги ручного коригування швидкості навчання під час процесу навчання. Окрім того, даний алгоритм зберігає експоненціально спадну середню попередніх градієнтів. Дана ідея дуже схожа на імпульс:

$$m_i = \beta_1 m_{i-1} + (1 - \beta_1) \frac{dLoss}{dW}$$

$$v_i = \beta_2 v_{i-1} + (1 - \beta_2) \left(\frac{dLoss}{dW} \right)^2$$

m_i та v_i - є ковзними середніми, які оцінюють перші два моменти градієнтів на параметр. Параметри β_1 , β_2 контролюють експоненційну швидкість спаду цих середніх. Однак, оскільки ці середні значення ініціалізуються нульовими векторами,

оцінки будуть зміщені до нуля, особливо під час початкових кроків. Щоб протидіяти цьому, оцінки коригуються до:

$$\hat{m}_i = \frac{m_i}{1 - \beta_1^i}$$

$$\hat{v}_i = \frac{v_i}{1 - \beta_2^i}$$

$$\mathbb{W}_{i+1} = \mathbb{W}_i - \frac{\eta}{\sqrt{\hat{v}_i} + \epsilon} \hat{m}_i$$

Потім ці оцінки, виправлені за зміщенням, використовуються для нормалізації градієнтів і множаться на глобальну швидкість навчання для оновлення ваг.

2.4 Висновок до розділу 2

В даному розділі було проаналізовано трьох лідерів, які надають можливість використання хмарних обчислень, а саме: Amazon, Microsoft Azure та Google Cloud Platform. Було описано основні переваги і недоліки, кожної. По всім критеріям AWS є поточним лідером ринку з точки зору потужності та обслуговування. Але його конкуренти також швидко розвиваються.

Зрештою, звичайно, все зводиться до конкретного випадку використання. Оскільки ринок зростає, більшість підприємств шукають багато хмарні стратегії, щоб використовувати переваги кожного постачальника хмарних послуг. Тобто не прив'язуватись до одного постачальника. Також був розглянутий один з можливих модулів для роботи з фото, а саме можливість кольоризації чорно-білих зображень. Було проведено опис згорткової нейронної мережі та самого алгоритму.

3 ПРОЕКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Опис завдання

Для реалізації власного архітектурного рішення при побудові програмного забезпечення для обробки оцифрованих фото було обрано такий стек технологій як: Auth0, Amazon Route 53, AWS WAF, Amazon CloudFront, Amazon S3, Amazon API Gateway, AWS Lambda, Amazon DynamoDB, Amazon SQS.

В свою чергу запропоноване програмне забезпечення підпорядковується наступним функціональним вимогам:

- реєстрація користувача з можливістю реєстрації через наступні сервіси: Google, Twitter, GitHub;
- можливість реєстрації компанії;
- можливість вибору пакету послуг для обробки зображень для компаній;
- можливість використання для оплати вибраного пакету послуг через платіжну систему Stripe;
- можливість компанії створювати користувачів через відправку запрошення на пошту;
- розподілення доступу до фото за правами та можливість створювати папки з фото;
- можливість автоматичної відправки щойно створеної картинки на API;
- можливість автоматичної відправки архіву з зображеннями на API;
- можливість кольоризації зображення.

Основі нефункціональні вимоги:

- можливість кешування зображення;
- легке горизонтальне масштабування;
- зменшення ресурсів для підтримки роботоспроможності серверу в залежності від навантаження;

– кроссплатформеність.

3.2 Використання Serverless

Serverless – це безсерверная архітектура для додатку. Основа даної архітектури полягає в розробці мікросервісів або лямбда функції. Вони виконують визначену логіку та запускаються на логічних контейнерах.

Відразу можна виділити основні переваги:

- відсутність апаратної частини, тобто серверу на якому розміщений додаток;
- відсутність прямого контакту та взаємодії і самого адміністрування серверної частини;
- дуже велика можливість горизонтального зростання самого проекту;
- оплата за використаний час CPU;
- прогон або розігрів контейнера;
- якщо описати основні недоліки, то можна виділити наступні;
- відсутність можливості чіткого контролю самого контейнера, тобто немає інформації як і де він запускається;
- важко спочатку зрозуміти поняття цілісності додатку в порівнянні з монолітом.

За 2021 рік відсоток використання без серверних виріс у півтора рази. Серед компаній, які вже внедрили технологію в свої сервіси, такі великі ринку як Twitter, PayPal, Mobile, Coca-Cola [20]. При цьому потрібно розуміти, що serverless — не панацея, інструмент для вирішення певного кола задач: зменшити ресурси, на льоту обробити дані, легка можливість роботи з Jira, Git.

Припустимо, є сервіс, на який приходить 100 осіб. Під нього стоїть сервер з віртуальною машиною із слабким залізом. Періодично навантаження обслуговування збільшується в багато разів і це призводить до того, що слабке залізо не справляється.

3.3 Розробка програмного забезпечення

Дане програмне забезпечення надає можливість редагувати та перетворювати зображення - представникам різноманітних компаній. Основна ідея полягає в тому, що компанії шукають можливість інтегрування своєї системи для того, щоб зменшити навантаження на обробці та зберіганню зображень, що дасть можливість позбутись проблеми горизонтального масштабування.

Для таких компаній, які шукають подібної можливості буде представлена інформативна сторінка з можливими тарифами для підписки. В залежності від потреб, компанія може обрати між наступними тарифами:

- обрати визначену кількість зображень для обробки незалежно від розміру, скільки займає саме зображення;
- безлімітний тариф, який буде включати обов'язкову оплату за рік використання.

Після реєстрації компанії в системі та вибору тарифу – необхідно сплатити сам тариф. Оплату можна з легкістю зробити через платіжну систему Stripe, яка буде використовуватись в даному програмному забезпеченні [35]. При аутентифікації даних власник компанії потрапляє в особистий кабінет де він повинен ввести всі необхідні серверні налаштування для коректної роботи. У компанії є можливість створювати користувачів, які будуть закріплені за компанією. Дані користувачі можуть редагувати набори зображень, кольоризувати при необхідності тощо.

Для побудови даного програмного забезпечення були використані технології, які описані в пункті 3.1. та запропонована наступна архітектура (рис. 3.1).

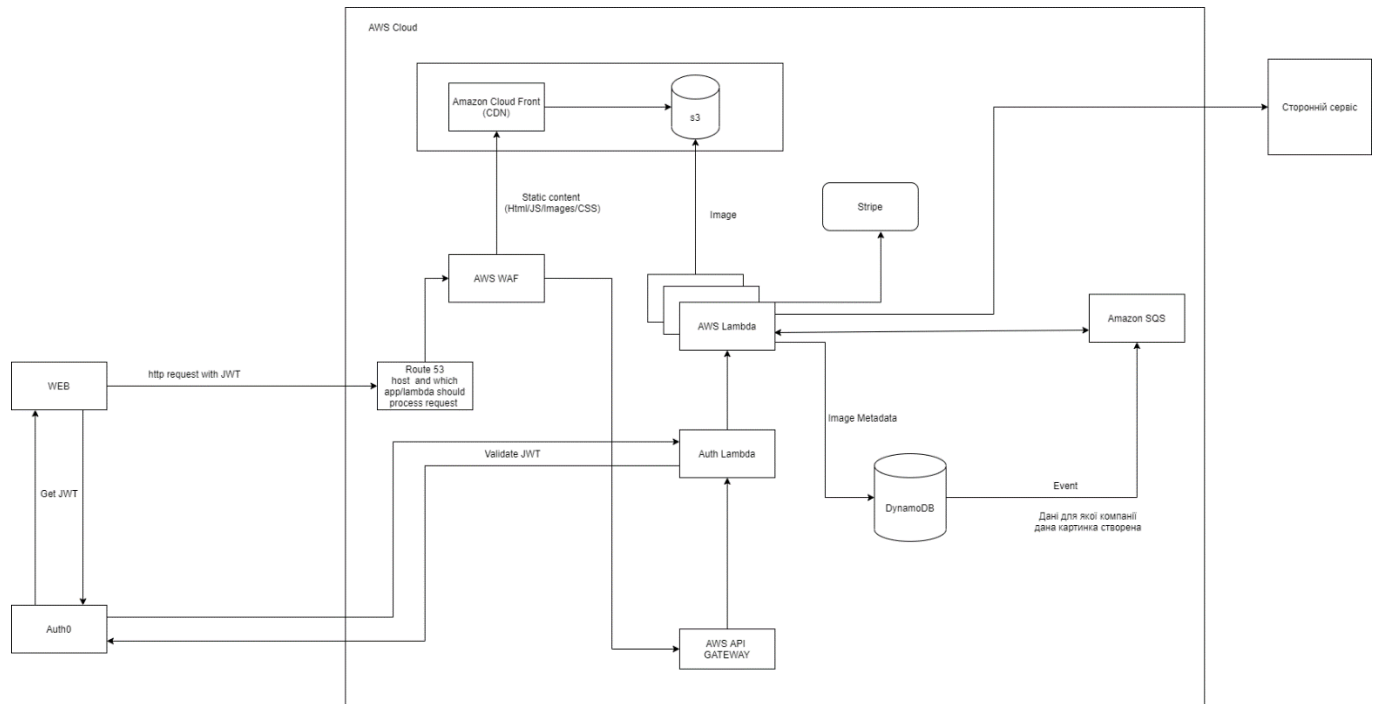


Рисунок 3.1 — Архітектура додатку

Amazon Route 53 – це хмарний веб-сервіс системи доменних імен, що може легко масштабуватись. Розробники веб-сервісів використовують даний сервіс, як виключно надійний та економічний спосіб перенаправлення кінцевих користувачів до додатків за рахунок перетворення доменних імен наприклад, `www.myservice.com` у формат цифрових IP-адрес [32]. Сервіс Amazon Route 53 надсилає запити користувачів до інфраструктури AWS, наприклад до інстансів Amazon EC2, балансувальників навантаження Elastic Load Balancing або кошиків Amazon S3. Крім того, його також використовують для перенаправлення користувачів на інфраструктуру за межами AWS. Даний хмарний сервіс можна налаштувати для перевірки працездатності DNS, а потім безперервно відстежувати та керувати відновленням програм через контролер Route 53 Application Recovery Controller [33].

Є можливість використати Traffic Flow і керувати глобальним трафіком, використовуючи різні типи маршрутизації. Поєднуючи маршрутизацію на базі затримки з урахуванням географічної близькості з'являється можливість перекидання

сервісу DNS. Тобто це означає, що можна створити відмовостійкі архітектури з низькою затримкою. Для того, щоб вести керування маршрутизацією користувачів до адрес додатків як в рамках одного регіону AWS, так і при розподілі трафіку по всьому світу, AWS надає візуальний редактор [34]. Також в сервісі є можливість зареєстрації доменних імен. На сервісах DNS буде автоматично налаштоване після покупки домену. При проходженні реєстрації, використовується Auth0 для отримання токена.

AWS WAF – це брандмауер для додатка. Він який дозволяє захистити його від поширених мережевих експлойтів та ботів. Дані загрози здатні вплинути на доступність, створити загрозу безпеці або задіяти надмірну кількість ресурсів програмного забезпечення. WAF надає повний контроль над тим, яким саме чином трафік досягатиме додатків користувача [30]. Тобто за допомогою даного сервісу описуються правила безпеки, які контролюватимуть трафік від ботів і блокуватимуть такі атаки, такі як SQL-ін'єкції або міжсайтовий скриптинг.

Також можна налаштувати правила фільтрації для певних шаблонів трафіку. Ви можете швидко розпочати роботу, використовуючи керовані правила для AWS WAF, попередньо вже налаштовано основні ризиків безпеки та автоматизовані боти, які споживають зайві ресурси, всі вони відразу відхиляються [31]. Ці правила регулярно оновлюються у міру виникнення нових проблем. AWS WAF пропонує повнофункціональний API, що дозволяє автоматизувати процеси створення, розгортання та обслуговування правил безпеки.

Amazon CloudFront – це хмарний веб-сервіс, який пришвидшує розповсюдження вашим користувачам статичного та динамічного веб-вмісту. До нього можна віднести .html, .js та файли зображень. CloudFront забезпечує доставку вмісту через мережу центрів обробки даних. Коли користувач робить запит на вміст, який ви обслуговуєте за допомогою CloudFront, запит направляється до граничного розташування, яке забезпечує найнижчу затримку [29]. В результаті вміст доставляється з найкращою можливою швидкістю. Якщо вміст уже знаходиться на

межі з найменшою затримкою, то CloudFront доставить його негайно, а у протилежному випадку CloudFront отримує його з джерела, який визначено з сегмента Amazon S3 або сервера HTTP, який було ідентифіковано як джерело остаточної версії вашого вмісту.

Amazon Simple Queue Service використовується як чергу повідомлень, яка дає змогу масштабувати мікросервіси та безсерверні програми. Використовуючи SQS, з'являється можливість отримувати, зберігати та надсилати повідомлення між компонентами на будь-якому рівні. Тобто даний сервіс використовується як черга по відправленню зображень [27]. Адже у випадку при пересилці великої кількості картинок може статись порушення пропускнуої здатності. Якщо виділити основні переваги, то отримаємо: безпека, довговічність, надійність, масштабованість.

Amazon S3 — використовується для зберігання об'єктів. Вона пропонує провідні технології що забезпечує наступні можливості, а саме: масштабованість, доступність даних, безпеку та високу продуктивність. Використання ефективного способу зберігання та упорядкування даних, під конкретні вимоги бізнесу дає великі можливості. Обумовленим використанням даної служби є те, що S3 є єдиною службою зберігання, яка має можливість заблокувати публічний доступ. Тобто доступ блокується до всіх об'єктів на рівні сегмента або облікового запису. Для цього достатньо використати S3 Block Public Access [26]. А отримання доступу до Amazon S3 буде відбуватись безпосередньо як приватна кінцева точка у віртуальній мережі за допомогою AWS PrivateLink. Він забезпечує приватне підключення між службами AWS і локальними мережами. Тобто використання AWS PrivateLink не відкриває ваш трафік у загальнодоступному доступі в мережу інтернет, а кінцеві точки інтерфейсу підключають відразу до служб [25]. Забезпечуючи живлення кінцевих точок, він забезпечує високий рівень безпеки та продуктивності. В результаті кінцевий юзер отримує лінки на зображення які будуть знаходитись в S3.

Для підтримки інтеграції даного сервіса з іншими продуктами, з боку продукту потрібно буде реалізувати дві кінцеві точки зв'язку для коректної роботи.

Перша кінцева точка буде використовуватись для отримання одного обробленого зображення, для цього потрібно вказати наступні параметри: `url`, `access_token`, `created_at`, описана на рис 3.2.

The screenshot displays the Swagger UI for a POST endpoint `/image`. The endpoint is described as "Add a new image to the store". A "Try it out" button is visible in the top right. The "Parameters" section shows a required `body` parameter of type `object` (body). The description for the body is "Image object that needs to be added to the store". An example value is provided in a dark box:

```
{
  "url": "s3uploader-4223newphoto",
  "access_token": "f616432f6d3124e6e0fa29d45818848de94267c747ac20e3a4f5f90d00195da39d2d5f26d218f4211f538",
  "create_at": "2021/11/01"
}
```

The parameter content type is set to `application/json`. The "Responses" section shows a `201` response with the description "Must returns 200 success response". The response content type is also set to `application/json`.

Рисунок 3.2 — Опис кінцевої точки зв'язку для отримання зображення

Друга кінцева точка використовується для отримання архіву опрацьованих зображень (рис. 3.3).

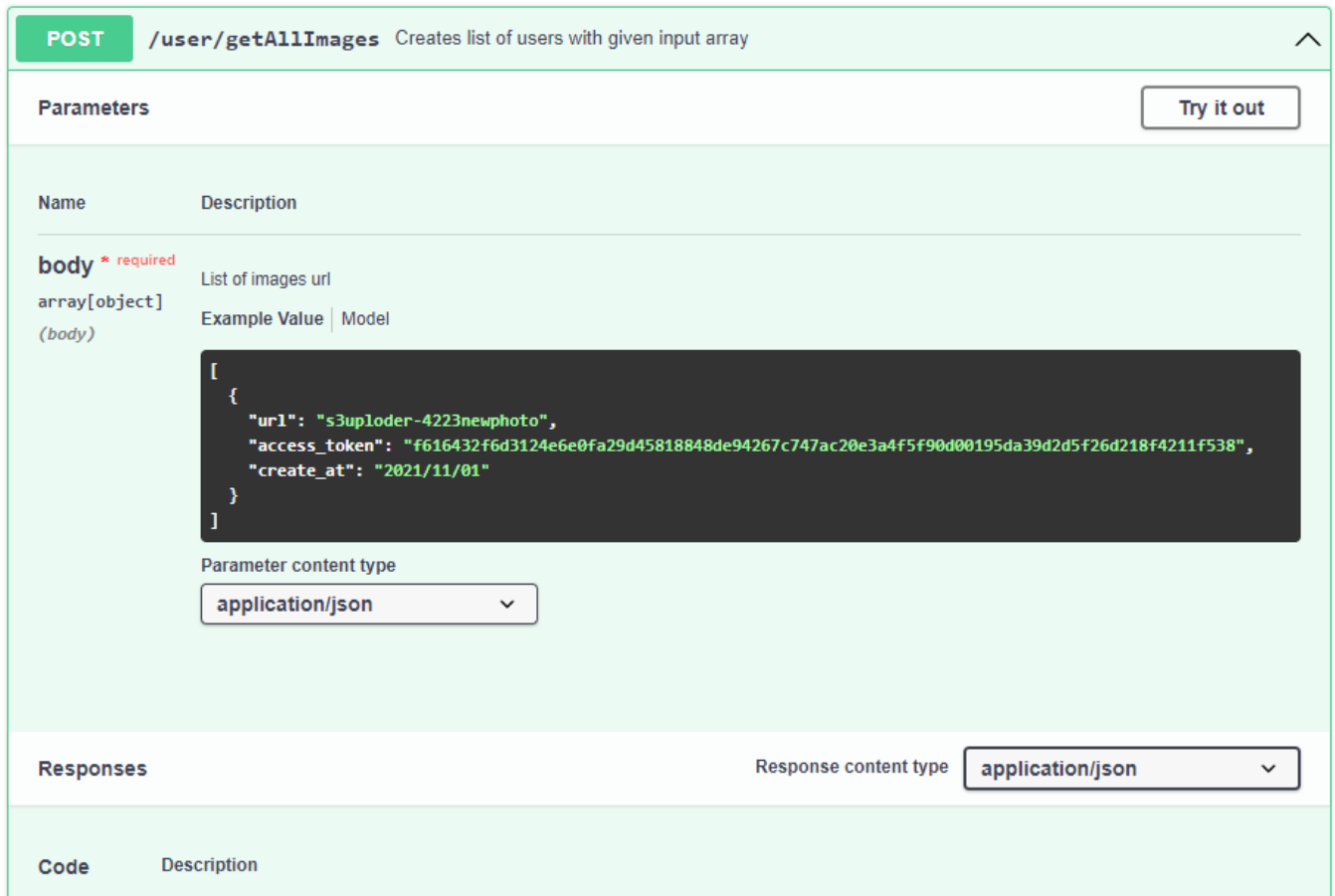


Рисунок 3.3 — Опис кінцевої точки зв'язку для отримання архіву зображень

Для опису кінцевих точок використано Swagger [28].

3.4 Аналіз безпеки даних користувача

Для забезпечення безпеки та коректної взаємодії з сторонніми сервісами було використано Auth0. Auth0 - це інструмент для аутентифікації як сервіс, він спрощує реалізацію функцій, які пов'язані із аутентифікацією для додатка. Даний сервіс легко забезпечує: єдина точка входу, багатофакторна автентифікація, вхід до системи без пароля та управління користувачами [16].

Аутентифікація на основі токенів в останні роки стала дуже популярна через поширення односторінкових додатків, API та інтернет речей. Найчастіше як токени використовуються Json Web Tokens [24]. Під час аутентифікації на основі токенів стан

не відстежуються. Не буде зберігатись інформація про користувача на сервері або сесії використані для клієнтів.

Процедура аутентифікації на основі JWT токенів, зображена на рис 3.4 й відбувається в такі етапи:

- користувач вводить логін та пароль;
- сервер перевіряє їх та повертає JWT токен, який може містити метадані: `user_id`, дозволи;
- токен зберігається на стороні клієнта, найчастіше в локальному сховищі, але може лежати і в сховищі сесій;
- наступні запити до сервера зазвичай містять цей токен як додатковий заголовок авторизації у вигляді `Bearer`;
- сервер розшифровує JWT, якщо токен вірний, сервер обробляє запит;
- коли користувач виходить із системи, токен на стороні клієнта знищується.

Можливість мати єдиної точки входу. Якщо звернути увагу, то при проходженні входу в браузері в якомусь Google-сервісі, наприклад Gmail, а потім йдеш на інший Google-сервіс, там не доводиться логінитися. Відбувається автоматичне отримання доступу до всіх сервісів компанії. Вони автентифікація користувача у всіх продуктах після єдиного входу. Такий підхід називається єдиною точкою входу.

Реалізувати даний підхід можна по-різному. Перше – це використовувати центральний сервіс для оркестрації єдиного входу між кількома сервісами. Коли користувач логіниться, сервіс створює куку, яка зберігається за користувачем, коли той ходить по сервісах, що належать компанії. Це працює наступним чином:

- користувач входить до одного із сервісів;
- користувач отримує генерований токен;
- користувач йде до іншого продукту;
- користувач знову перенаправляється в назад до першого сервісу;

– сервіс бачить, що користувачеві вже присвоєно токен і перенаправляє користувача.

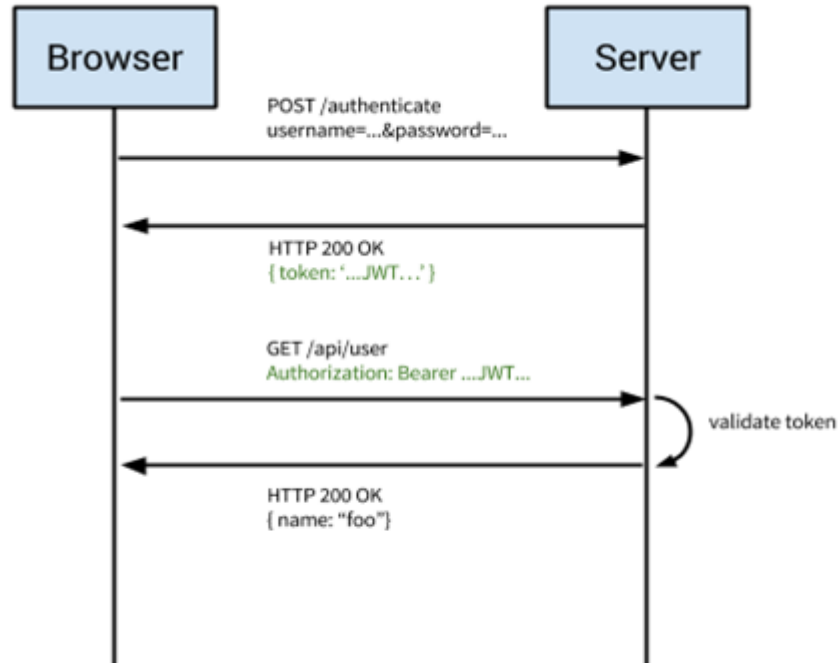


Рисунок 3.4 — Схематичне зображення видачі токена для авторизації [16]

3.5 Опис таблиць в DynamoDB

В даному проєкті для коректної роботи самого сервісу буде використана нереляційна база даних – DynamoDB. Дана база даних є повністю керована та безсерверна. Вона відноситься до NoSQL баз і побудована на основі пари ключ-значення, тому вона має гнучку схему, тобто кожен рядок у будь-який час може містити будь-яку кількість стовпців [23]. Це дозволяє легко адаптувати таблиці зі зміною вимог вашого бізнесу та позбавляє необхідності перевизначати схему таблиці та мігрувати, як в реляційних базах даних.

Найчастіше використовується для запуску високо виробничих систем для будь-якого масштабу. Вона пропонує: вбудований захист, неперервне резервне копіювання, автоматичну реплікацію в декількох регіонах, кешування в пам'яті та різноманітні інструменти для експорту даних:

Виділимо основні переваги:

- пара ключ-значення та використання документної моделі даних;
- продуктивність в масштабі;
- безсерверна;
- dynamoDB Accelerator відповідає за скорочення часу відповіді до кількох мікросекунд, на основі кешу пам'яті;
- автоматизована глобальна реплікація глобальних таблиць;
- вбудована підтримка ACID;
- шифрування всіх даних при зберіганні.

У табл. 3.1-3.3 описана структура таблиць бази даних.

Таблиця 3.1 — Опис таблиці Company

Назва елементів таблиці	Опис	Тип даних
company_id	Ід компанії	UUID
company_name	Назва компанії	String
subscription	Тип пакету підписки	String
created_at	Дата створення публікації	String
service_settings	Налаштування сервіса для коректної роботи	json
access_token	Токен для стронього сервісу в якому міститься інформація про компанію	String

Таблиця 3.1 зберігає інформацію про компанію. Елемент таблиці `service_settings` виступає в форматі `json`. Даний `json` включає в себе наступні поля, а саме:

- шлях на кінцеву точку зв'язку стороннього сервісу для відправки обробленого зображення;
- шлях на кінцеву точку зв'язку стороннього сервісу для відправки обробленого архіву з зображеннями;
- `boolean` флаг для підтвердження автоматичної відправки на ендпоінт опрацьованого зображення;
- токен (`access_token`) для авторизації запитів на сторонній сервіс.

Таблиця 3.2 — Опис таблиці `Image`

Назва елементів таблиці	Опис	Тип даних
<code>image_id</code>	Ід картинки	UUID
<code>folder_name</code>	Global secondary index	String
<code>source_url</code>	Шлях до картинок в s3	String
<code>created_at</code>	Дана створення зображення	String

Таблиця 3.3 — Опис таблиці `User`

Назва елементів таблиці	Опис	Тип даних
<code>email</code>	Пошта	UUID

Продовження таблиці 3.3

company_id	Global secondary index	String
name	Ім'я користувача	String
surname	Фамілія користувача	String
last_logged_at	Дата останнього відвідування	String

3.6 Демонстрація програмного забезпечення

Для початку роботи потрібно пройти реєстрацію та заповнити всі необхідні поля вводу, як це зображено на рис 3.5.

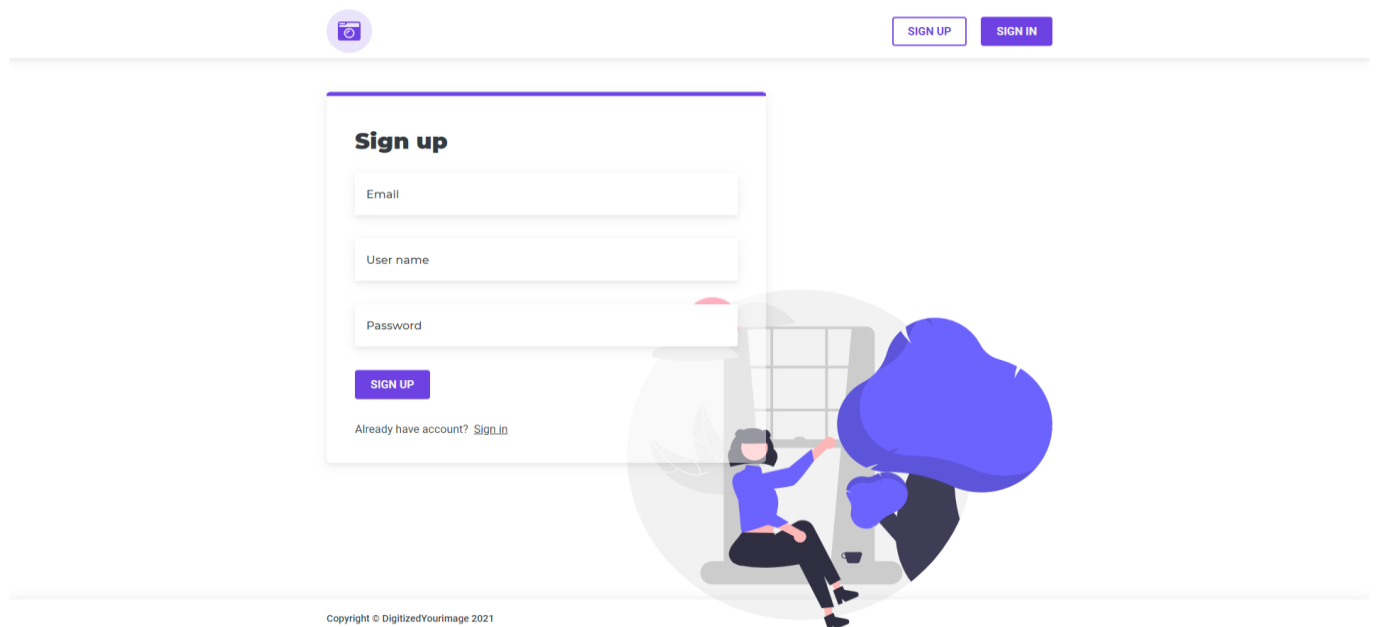


Рисунок 3.5 — Реєстрація в системі

Спроектоване програмне забезпечення надає можливість тісної інтеграції з продуктами, яким необхідна можливість в редагуванні зображення. Після вибору та

оплати користувацького пакету власники компаній можуть реєструвати своїх співробітників та переглядати статистику як це зображено на рис. 3.6.

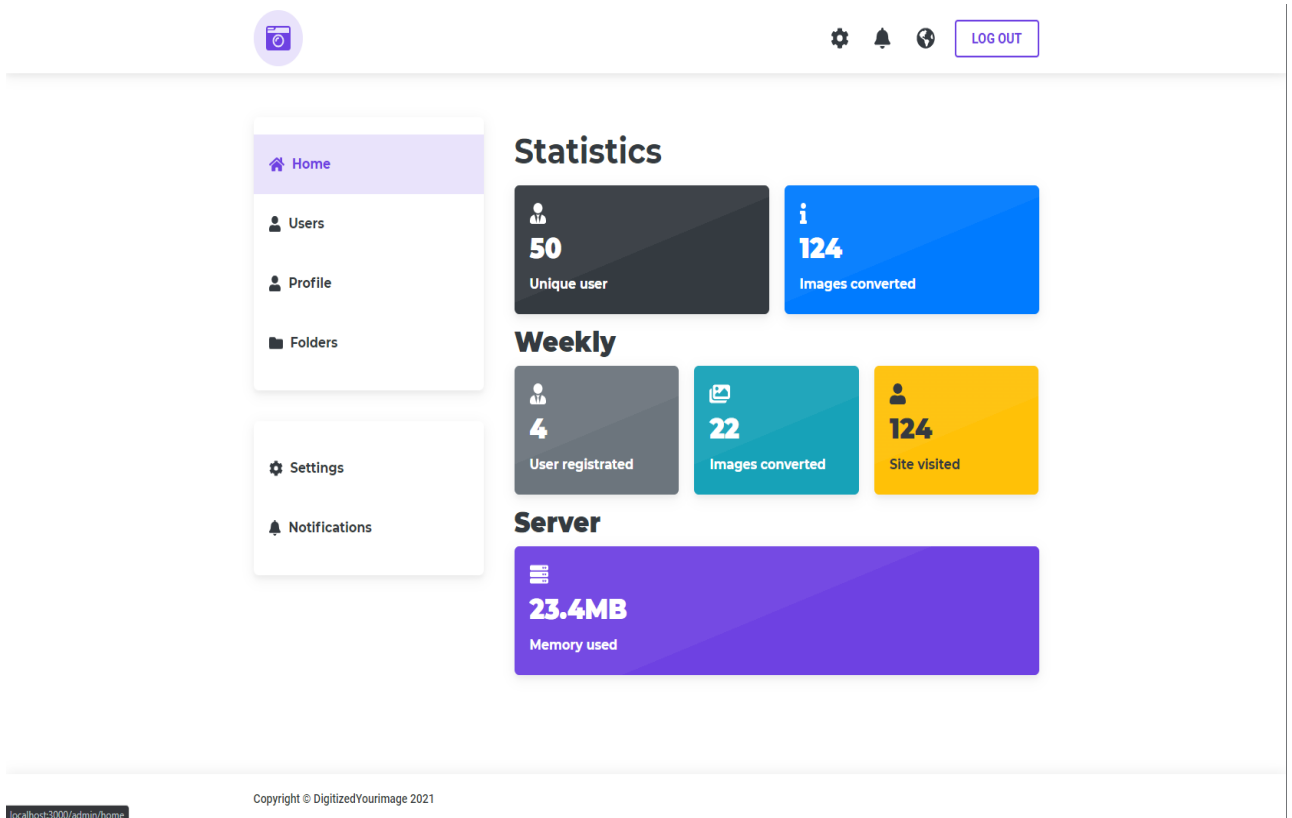
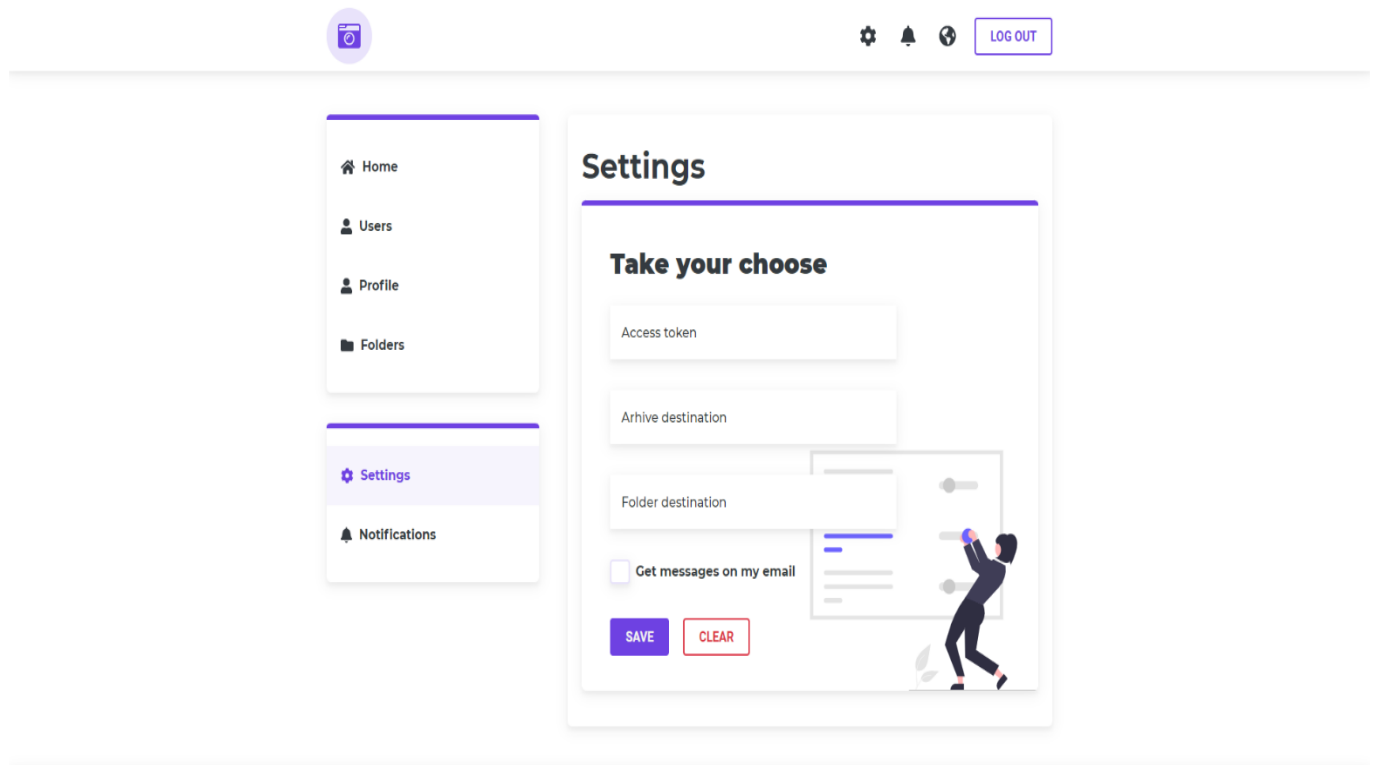


Рисунок 3.6 — Особистий кабінет керівника компанії

Все це забезпечує автоматичне розподілення ролей та чіткі вимоги до певної ролі. Компанія може переглядати інформаційну лендінг сторінку з основним функціоналом та обирати для своїх потреб той тарифний план, який їй буде вигідно.

Так як даний сервіс забезпечує можливість інтеграції з боку стороннього продукту достатньо буде описати дві кінцеві точки для отримання зображень. Для цього необхідно власнику компанії вказати наступні налаштування, які зображені на рис. 3.7. Так як даний сервіс спроектований на хмарних технологіях він надає можливість горизонтального розширення з мінімальними витратами на підтримку.

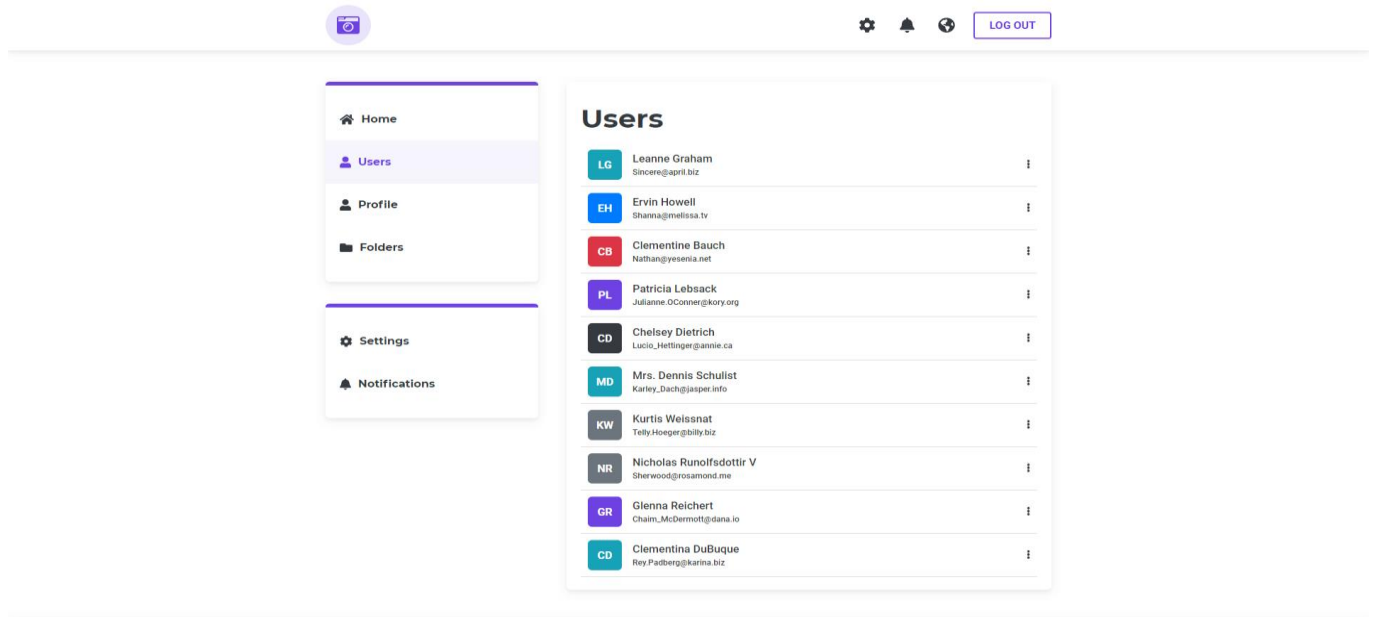
Тобто при навантаженні на статичний сервер, а саме при збільшенні користувачів та їх запитів, його потрібно розширювати. А також окрім витрат на апаратну частину необхідна наявність девопсів, все це веде до матеріальних витрат.



Copyright © DigitizedYourImage 2021

Рисунок 3.7 — Налаштування для керівника компанії

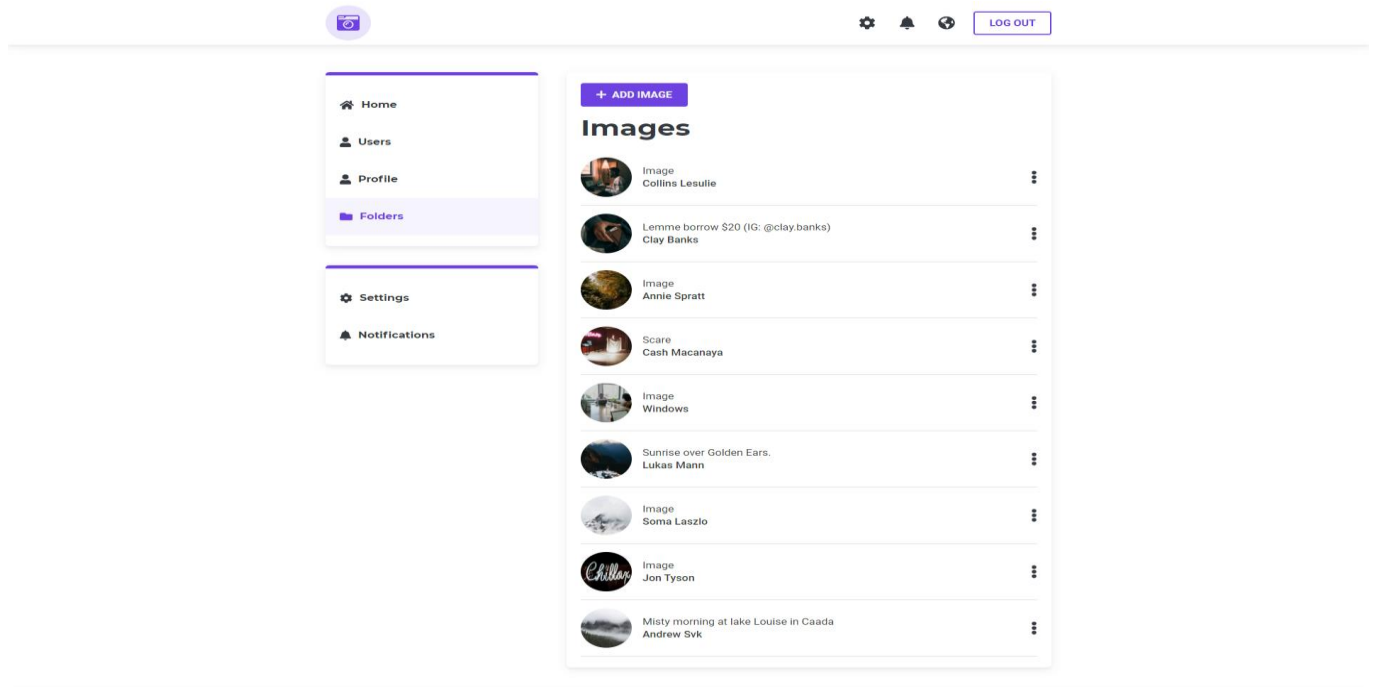
Також є перегляд списку працівників компанії та можливість відвідування профілю працівника, як зображено на рис. 3.8.



Copyright © DigitizedYourImage 2021

Рисунок 3.8 — Список працівників компанії

В свою чергу керівник реєструє працівників, які мають свій особистий кабінет, можуть загрузити зображення до папок, редагувати їх та відправляти дані зображення на сторонній сервіс, профіль працівника зображений на рис. 3.9.



Copyright © DigitizedYourImage 2021

Рисунок 3.9 — Профіль працівника компанії

3.7 Експериментальні дослідження

Для перевірки роботоспроможності програмного забезпечення було створено кінцеву точку для отримання зображень в якості стороннього продукту, опис кінцевих точок було зроблено в розділі 3.3. Для наглядної демонстрації було створено додаткову кінцеву точку, яка приймає на стороні програмного забезпечення набір зображень та токен самої компанії. Експериментальні результати наведені в таблиці 3.4.

Таблиця 3.4 — Опис експериментальних досліджень

Кількість зображень	Час опрацювання запиту та відповіді	Кольорове зображення
100	0.15 секунд	Так
1000	1,68 секунд	Так
10000	17,02 секунд	Так
100	0.09 секунд	Ні
1000	1,1 секунд	Ні
10000	13,4 секунд	Ні

3.8 Оцінка ефективності запропонованого рішення

Якщо розглядати дане програмне забезпечення, як стартап, потрібно розуміти, що на початкових етапах стартап може не приносити прибуток. Виділення інвестиції для купівлі свого серверу будуть фінансовим ризиком для самої стартап компанії. Створення програмного забезпечення на основі хмарних технологій забезпечить майже безкоштовне користування даним сервісом на початкових етапах. За допомогою конфігураційного калькулятора AWS можна оцінити скільки необхідно витрат для утримання подібного серверу, даний сервіс зображено на рис. 3.10.

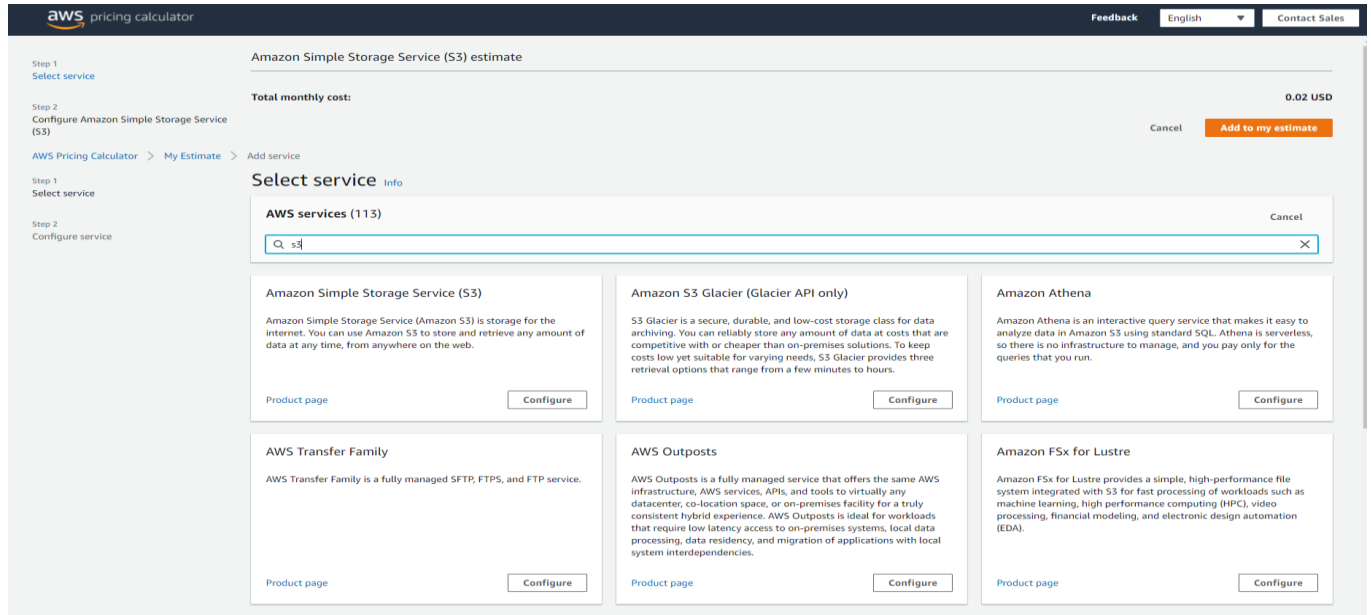


Рисунок 3.10 — Калькулятор для підрахунку витрат AWS

Таблиця 3.5 — Опис витрат на місяць для підтримки сервісу

Опис AWS сервісів	Менше 1000 запитів	1000000 запитів	понад 2000000 запитів
S3	0,0026 грн	196,81 грн	від 300 грн
DynamoDb	0,0023 грн	163,14 грн	від 150 грн
Amazon SQS	0,0051 грн	103,64 грн	від 260 грн
Amazon Cloud Front	0,0047 грн	109 грн	від 200 грн
AWS WAF	0,001 грн	107,52 грн	від 200 грн
AWS Lambda/Auth Lambda	0,0017 грн	68,97 грн	від 170 грн
AWS API Gateway	0,003 грн	83,24 грн	від 230 грн
Route 53	0,001 грн	107,89 грн	від 180 грн
Сума	0,00664 грн	940 грн	Від 1690 грн

Дані витрати були сформовані за допомогою сервісу який надає AWS з урахуванням середньо статистичних вимог, приклад підрахунку витрат для S3 зображений на рис 3.11.

aws pricing calculator Зворотній зв'язок англійська Зв'яжіться з відділом продажів

Стандарт S3

▼ Стандарт S3 інформація

Наведені нижче розрахунки не включають знижки безкоштовного рівня.

S3 Стандартне сховище
 ГБ на місяць

Запити PUT, COPY, POST, LIST до S3 Standard

GET, SELECT та всі інші запити від S3 Standard

Дані, повернуті S3 Select
 ГБ на місяць

Дані, скановані за допомогою S3 Select
 ГБ на місяць

▼ Показати розрахунки

Рівнева ціна за: 500 ГБ
 500 ГБ x 0,0230000000 USD = 11,50 USD
 Загальна вартість рівня = 11 5000 доларів США (стандартна вартість зберігання S3)
 100 000 запитів PUT для зберігання S3 x 0,000005 USD за запит = 0,50 USD (вартість стандартних запитів S3 PUT)
 1 000 000 запитів GET на місяць x 0,000004 USD за запит = 0,40 USD (вартість запитів S3 Standard GET)
 500 ГБ x 0,0007 USD = 0,35 USD (S3 виберть вартість повернення)
 500 ГБ x 0,002 USD = 1,00 USD (S3 виберть вартість сканування)
 11,50 USD + 0,40 USD + 5,00 USD + 0,35 USD + 1,00 USD = 18,25 USD (усього S3 Standard Storage, запити даних, S3 вибрати вартість)
S3 Стандартна вартість (щомісячно): 18,25 USD

Рисунок 3.11 — Приклад розрахунку втрат для S3

Підсумуємо функціональні можливості спроектованого програмного забезпечення в порівнянні з двома головними конкурентами.

Таблиця 3.6 – Функціональні можливості

Параметри	Colorize B&W Photos	Розроблене програмне забезпечення	PhotoshopOnline
Колоризація	+	+	-
Вільний доступ без використання VPN	+	+	+

Продовження таблиці 3.6

Наявність базового набору графічних інструментів	-	+	+
Хмарна архітектура	-	+	-
Наявність особистого кабінету	-	+	-
Наявність бази користувачів	-	+	-
Можливість перегляду попередніх зображень та папок	-	+	-
Можливість інтеграції системи з іншими продуктами	-	+	-

Також потрібно виділи основні переваги спроектованого програмного забезпечення у в порівнянні з MVC архітектурою.

Таблиця 3.7 – Переваги використання архітектурного рішення

Переваги	Пояснення
Масштабованість	Вирішення проблеми з масштабованістю залежно від поточного навантаження. Тобто спроектована система автоматично збільшує чи зменшує пропускну здатність.

Продовження таблиці 3.7

Збільшення швидкості обробки зображення та даних в цілому	Затримка стає мінімальною за рахунок використання хмарних сервісів. В порівнянні з звичайною архітектурою MVC та фізичним сервером збільшення в 7%.
Стабільність системи	Так як відповідальність за обслуговування сервері лягає на AWS, вхідні дані на логуються та відбувається клонування даних для забезпечення надійності та стабільності. Зникає проблема у технічних збіях, які часто трапляються з фізичним сервером.

3.9 Висновки по розділу 3

В даному розділі було описано етап проектування архітектури програмного забезпечення обробки оцифрованих зображень з використанням хмарних технологій. Спроектовано та описано архітектурне рішення, технічні вимоги, зроблено аналіз витрат на використання AWS, надано структуру бази даних та описано функціональні та нефункціональні вимоги. Було зроблено аналіз оцінки ефективності запропонованого архітектурного рішення та наведено основні переваги даного архітектурного рішення.

4 МАРКЕТИНГОВИЙ АНАЛІЗ СТАРТАП-ПРОЄКТУ

4.1 Опис ідеї проєкту

Ідея даного проєкту полягає в тому, щоб створити програмне забезпечення з використанням хмарних технологій для обробки зображень з можливістю інтеграцій з іншими продуктами. Спроектвана архітектура забезпечує підвищення горизонтального масштабування та забезпечує зменшення ресурсів для підтримки серверу. В цілому дана система повинна підтримувати наступний функціонал:

- можливість реєстрації компанії;
- можливість вибору пакету послуг для обробки зображень для компанії;
- можливість використання для оплати вибраного пакету послуг через платіжну систему Stripe;
- усунення затемнень та шумів на зображенні;
- можливість компанії створювати користувачів через відправку запрошення на пошту;
- розподілення доступу до фото за правами та можливість створювати папки з фото;
- можливість автоматичної відправки щойно створеної картинки на API;
- можливість автоматичної відправки архіву з зображеннями на API;
- можливість кольоризації зображення.

Таблиця 4.1 — Опис ідеї стартап-проєкту

Зміст ідеї	Напрямки застосування	Вигоди для користувача
Сервіс в якому можуть реєструватись компанії, обирати тарифи по редагуванню зображень, які будуть їм вигідні також кольоризувати чорно-білі зображення з можливістю зберігання та відправки на свій сервіс перетворені зображення.	Перетворення та редагування зображень як самостійного сервісу.	Вибір тарифу, який буде підходити їм особисто. Можливість опрацювання як окремих зображень так і цілого архіву з фото та автоматична відправка перетворених зображень на сторонній сервіс. Зменшення навантаження на сторонній сервіс по зберігання зображень.
	Інтеграція з сторонніми продуктами, яким необхідна можливість редагування зображень.	

Зробивши огляд ринку з існуючими аналогами, було зроблено висновок, що більшість систем представлені як звичайний фоторедактор без можливості розширення додатку та інтеграції.

Таблиця 4.2 — Опис техніко-економічної характеристики ідеї

№	Техніко-економічні характеристики ідеї
1	Зручність використання
2	Кросплатформність
3	Можливість інтеграції з сторонніми продуктами
4	Функціональні вимоги до самої системи

Таблиця 4.3 — Опис ідеї стартап-проєкту

№	Продукція конкурентів				W (слабка сторона)	N (нейтральна сторона)	S (сильна сторона)
	Мій проєкт	Colorize B&W Photos	Pho.to	Photosho pOnline			
1	Зручно	Зручно	Зручно	Зручно			+
2	Наявна	Наявна	Відсутня	Наявна		+	
3	Наявна	Відсутня	Відсутня	Відсутня			+
4	Високі	Низькі	Низькі	Високі		+	

Ідея запропонованого програмного забезпечення є актуальною в даний час, можна виділити багато переваг для користувачів даної системи. Зробивши аналіз та опис техніко-економічні характеристики, сильні та слабкі сторони можна зробити висновок, що проєкт має перспективу.

4.2 Технологічний аудит ідеї проєкту

Для аналізу технічного аудиту ідеї даного проєкту, необхідно провести аудит технологій, за допомогою яких можна реалізувати описане програмне забезпечення. Першим кроком необхідно визначити технологічну здійсненність програмного забезпечення.

Описані технології є доступними та мають багато відкритої документації для подальшого створення продукту. Для створення даного програмного забезпечення з використанням даної технології необхідно: персональний комп'ютер, робоче середовище для написання програмного коду та доступ до мережу інтернет.

Таблиця 4.4 — Технологічна здійсненність ідеї проєкту

№	Ідея проєкту	Технології її реалізації	Наявність технологій	Доступність технологій
1	Програмне забезпечення для обробки оцифрованих зображень	Використання хмарних технологій	Розроблені фреймворки для побудови системи	Повністю відкрита та наявна документація
2	Побудова API для розфарбування чорно-білих оцифрованих фото	Використання згорткових нейронних мереж	Розроблені бібліотеки для роботи з нейронними мережами	Повністю відкрита та наявна документація
<i>Обрана технологія реалізації ідеї проєкту: 1</i>				

Висновок: технологічна реалізація продукту – можлива, вибрана технологія №1. При використанні даної технології можна отримати перспективний продукт, який можна буде при необхідності з легкістю інтегрувати з іншими системами і мати багато переваг.

4.3 Аналіз ринкових можливостей запуску стартап-проєкту

Аналіз ринкових можливостей, допомагає позбутися ринкових загроз та спланувати напрями розвитку проєкту з урахуванням стану ринкового середовища.

Для того, щоб забезпечити активний попит на продукт потрібно провести аналіз ринку.

Якщо врахувати аналоги, які пропонує ринок, проаналізувати весь доступний функціонал та переглянути тенденції останніх років на автоматизацію та диджеталазацію процесів з урахуванням карантинних обмежень – можемо зробити висновок, що ринок та ідея для створення стартап-продукту є привабливою для подальшого аналізу.

Таблиця 4.5 — Попередня характеристика потенційного ринку

№	Показники стану ринку	Характеристика
1	Кількість головних гравців, од	3
2	Загальний обсяг продаж, грн./ум.од	1000
3	Динаміка ринку	Темпи розвитку світової економіки є позитивними, на додаток до цього, з ознаками збільшення росту.
4	Наявність обмежень для входу	Відсутні. Так як після аналізу аналогів – вони виявились самостійними одиницями без можливістю інтеграцій з іншими системами
5	Специфічні вимоги до стандартизації та сертифікації	Відсутні
6	Середня норма рентабельності в галузі або по ринку, %	70%

Висновок: враховуючи, що кількість головних гравців по ринку тільки 3, але й вони позиціонують себе як незалежні сервіси по обробці зображень, зростаючу динаміку ринку з урахуванням карантинних обмежень, середню норму рентабельності, то можна зробити висновок, що на даний момент, ринок для входження стартап-продукту є привабливим.

Таблиця 4.6 — Характеристика потенційних клієнтів стартап-проєкту

№	Потреба, що формує ринок	Цільова аудиторія	Відмінності у поведінці цільових груп клієнтів	Вимоги споживачів до товару
1	Потреба в можливості редагування зображень при інтегруванні з іншими системами	Працевлаштоване населення	Стартап буде використовуватись для економії часу на автоматичному відправленні зображення після редагування	Легкий та зрозумілий інтерфейс, локалізація, використання на різних платформах
2	Потреба в горизонтальному масштабуванні сервіса а також зменшення ресурсів для підтримки серверу	Власники сервісів, які хочуть зберегти кошти на розробці подібного функціоналу, отримавши можливість зайняти конкуруючу позицію на ринку	Стартап буде використовуватись для економії на ресурсах підтримки серверу та забезпечення горизонтального масштабування	можливість редагування зображень, перегляд попередніх зображень, різномірівнева система доступів до.

Продовження таблиці 4.6

3	Зручний інтерфейс, кросплатформність, надійність	Великі компанії Невеликі приватні компанії Державні установи Фізичні особи	Для всіх це зменшена ціна. Для великих компаній – скептичне ставлення до маловідомого продукту	зображень, можливість економії на зберіганні зображень в сторонньому сервісі.
---	--------------------------------------------------	-------------------------------------------------------------------------------------	-------------------------------------------------------------------------------------------------------	-------------------------------------------------------------------------------

Опис та аналіз характеристики потенційних клієнтів дозволяє зробити наступний висновок. За рахунок подальшого масштабування системи, розширення функціоналу та забезпечення в безпеці даних, можливо сказати, що дана система буде хорошим універсальним рішенням при інтеграції з іншими системами.

Таблиця 4.7 — Фактори загроз

№	Фактор	Зміст загрози	Можлива реакція компанії
1	Поява конкурентів	Наявність конкурентів котрі надають схожі рішення	Зменшення ціни на поставлену послугу. Розробка унікальних характеристик товару. Надання ліцензій на обслуговування.
2	Кошти на розробку та підтримку продукту	Закінчення грошей та недостатнє фінансування	Залучення додаткових інвесторів, мотивація роботи на перспективу

Продовження таблиці 4.7

3	Вихід аналогу	Вихід аналогу може призвести до знецінення ідеї та продукту в цілому	Проведення рекламної кампанії. Швидке захоплення більшої частини ринку. Акційні пропозиції для постійних користувачів.
4	Неправильно визначена цільова група для реклами	Тобто таргетация реклами неправильно визначена і реклама з'являється людям які не зацікавлені в цьому продукту	Вдосконалити рекламну кампанію правильно визначити таргет групу
5	Недостатня кількість реклами	Тобто мала кількість реклами в соцмережах і спеціальних онлайн форумах	Вдосконалити рекламну компанію збільшити кількість реклами

Таблиця 4.8 — Фактори можливостей

№	Фактор	Зміст можливості	Можлива реакція компанії
1	Новий продукт	Вихід на ринок, Зменшення монополії,	Розробка нової функціональності. Вихід нової продукції на ринок.

Продовження таблиці 4.8

2	Збір інформації з потенційних конкурентів та вихід аналогу	Надати продукт з певними характеристиками та можливостями що відсутні у компаній конкурентів	Аналіз ринку та користувачів задля задоволення їх потреб та надання функціональності у найкоротші строки за ціну, котра є дешевшою ніж у продуктів-замінників.
3	Зворотній зв'язок від користувачів	Можливість отримання необхідної інформації для вдосконалення продукту	Наявність вхідних даних та реакція на них з боку команди розробників задля задоволення потреб кінцевих користувачів системи.

Таблиця 4.9 — Ступеневий аналіз конкуренції на ринку

№	Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)
1	Монополія	Один головний товаровиробник	Контроль цін щоб маневрувати між конкурентами

Продовження таблиці 4.9

2	За рівнем конкурентної боротьби: - світовий	Конкуренти по всьому світу	Вихід на ринок збуту продукту з клієнто-необхідною функціональністю; Налагодження маркетингу на основних Інтернет ресурсах задля охоплення великої кількості потенційних користувачів;
3	За галузевою ознакою: внутрішньогалузева	Конкуренція в сфері фінансових ринків	Розробка продукту який вирішує питання з прогнозуванням росту фінансового ринку
4	Конкуренція за видами товарів: -товарно-видова	Дана конкуренція – конкуренція між товарами одного виду.	Впровадження функціональності яка відсутня у товарів-замінників; Спрощення інтерфейсів; Надання підтримки.
5	За характером конкурентних переваг: -цінова та не цінова	Цінові переваги – точкова комерціалізація; Не цінова – надання функціональності, що відсутня у товарах-замінниках.	Надання платних ліцензій лише на критично важливу функціональність для клієнта з певним строком підтримки, що зазначена у відповідній ліцензії; Впровадження унікальної функціональності.

Таблиця 4.10 — Аналіз конкуренції в галузі за М. Портером

С к л а д о в і а н а л і з у	Прямі конкуренти в галузі	Потенційні конкуренти	Постачальники	Клієнти	Товари-замінники
	Невеликі інтерпрайс компанії	Монополія	Розміри проданого продукту	1)Високий відсоток прогнозувань; 2)Наявність багатого функціоналу; 3)Дешева ціна;	Достатньо висока ціна
В и с н о в к и	Середня інтенсивність конкурентної боротьби стимулює вдосконалювати продукт	на даний момент ідея з інтеграцією з іншими продуктами є перспективною	Великі обсяги проданого продукту	Доступна ціна й необхідний функціонал, а також можливість зворотнього зв'язку	Обмеження відсутні

Після аналізу можливостей роботи на ринку з огляду на конкурентну ситуацію можна зробити наступний висновок. Кожний з існуючих продуктів не впливає у великій мірі на поточну ситуацію на ринку в цілому, адже в порівнянні з існуючими програмними забезпеченнями вони мають свою специфічну сферу використання та свої переваги та недоліки сторони [17]. З цього можна зробити висновок, що

проектування, розробка даного продукту та вихід на ринок є можливою і реалізованою задачею.

Для виходу на ринок продукт повинен мати функціонал, який буде приваблювати користувачів мати, мати зрозумілий та доступний інтерфейс та конкуренту швидкість роботи.

Таблиця 4.11 — Обґрунтування факторів конкурентоспроможності

№	Фактор конкурентоспроможності	Обґрунтування
1	Ціна	Вона буде трошки нижча ніж в аналогів спочатку для того щоб набрати користувачів
2	Кросплатформеність	Використання боту на різних платформах
3	Швидка та надійна робота серверу	Швидка робота серверу, адже кожна секунда це гроші та безпека, щоб користувачі не хвилювались за те що буде витік даних
4	Наявність всього необхідного інструменту для того щоб редагувати зображення	Наявність функціоналу по редагуванню зображення
5	Наявність можливості кольоризації та навпаки перетворення в чорно-біле зображення	Поєднання можливостей редагування і кольоризації зображень є хорошим кроком для створення універсальної системи.

Продовження таблиці 4.11

6	Приватність	Приватність інформація є важливим критерієм для спокійного використання користувачем
7	Технічна підтримка	Технічна підтримка буде працювати швидко та якісно, то це допоможе зберегти репутацію компанії
8	Можливість інтеграції з іншими продуктами	Автоматична відправка та створення простору для користувачів компанії для редагування зображення з можливістю відправки на інший сервіс

Таблиця 4.12 — Порівняльний аналіз сильних та слабких сторін системи по обробці зображень

№	Фактор конкурентоспроможності	Бали 1-20	Рейтинг товарів-конкурентів у порівнянні з запропонованим						
			-3	-2	-1	0	+1	+2	+3
1	Ціна	15			Наш	К3	К2	К1	
2	Кросплатформеність	14	К2	К1	К3		Наш		
3	Швидка та надійна робота серверу	19	К2		К1		К3		Наш
4	Наявність всього необхідного інструменту	18				К1	К2	Наш	К3

Продовження таблиці 4.12

5	Наявність можливості кольоризації та навпаки перетворення в чорно-біле зображення	16					Наш	К1	
6	Приватність	20	К1	К2	К3		Наш		
7	Технічна підтримка	16	К3	К2		К1			Наш
8	Можливість інтеграції з іншими продуктами	20							Наш

Наступним етапом ринкового аналізу для програмного забезпечення по обробці зображень є складання SWOT аналізу, який будується на основі виділених ринкових загроз, слабких та сильних сторін, можливостей. На основі даного аналізу формуються та розробляються альтернативи ринкової поведінки.

Таблиця 4.13 — SWOT аналіз стартап-проєкту

<p>Сильні сторони (S):</p> <ul style="list-style-type: none"> – горизонтальне масштабування; – зрозуміла архітектура; – ціна; – можливість редагування та кольоризації зображення в одному сервісі; – можливість інтеграції з іншими продуктами; 	<p>Слабкі сторони (W):</p> <ul style="list-style-type: none"> – тенденції ринку вдосконалення алгоритмів та нейронної мережі; – тривала розробка; – тренування нейронної мережі;
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Продовження таблиці 4.13

Можливості (О):	Загрози (Т):
<ul style="list-style-type: none"> – реєстрації компаній та вибору пакету послуг; – компанії реєструють користувачів; – можливість редагування та кольоризації зображення; – безпечний та горизонтально масштабований сервіс; – можливість інтеграції з іншими продуктами та автоматична відправка як одиничного так і архіву зображень; 	<ul style="list-style-type: none"> – взлом сервісу і отримання доступу до s3;

Таблиця 4.14 — Альтернативи ринкового впровадження стартап-проекту

№	Альтернатива (орієнтовний комплекс заходів) ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації
1	Безкоштовне надання можливості роботи з зображенням на обмежений термін	Головний ресурс – люди, даний ресурс - наявний	1 місяць
2	Реклама, рекламні інтеграції	Залучення власних коштів для реклами товару	1-3 місяців
3	Написання статей та опис товару на відомих ресурсах	Головний ресурс – час, даний ресурс - наявний	2-3 тижні

4.4 Розроблення ринкової стратегії проєкту

Таблиця 4.15 — Вибір цільових груп потенційних споживачів

№	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи (сегменту)	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
1	Великі компанії	Потребують	Попит є, у великих компаніях у випадку, коли вони хочуть отримати швидкий результат і зайняти більшу частину ринку	Майже відсутня	Помірно
2	Невеликі компанії	Потребують	Попит є	Майже відсутня	Помірно
3	Звичайні одноосібні користувачі	Потребують	Попит є, але їх будуть цікавити більш відомі продукти	Присутня	Помірно
Які цільові групи обрано: Всі					

Було обрано наступні цільові групи. Оскільки різниця між цільовими групами зовсім незначна, а також враховуючи той факт, що планується отримання прибутку якомога швидше, то доцільно враховувати усі цільові групи, тобто використовувати масовий маркетинг, пропонуючи стандартизовану програму з особливими можливостями.

Таблиця 4.16 — Визначення базової стратегії розвитку

Обрана альтернатива розвитку проєкту	Стратегія охоплення ринку	Ключові конкурентоспроможні позиції відповідно до обраної альтернативи	Базова стратегія розвитку
Надання функціональності що відсутня у товарів-замінників, підтримка клієнтів	Проведення таргетованої реклами, освітлення унікальної функціональності через інтернет ресурси та інші канали, формування лояльності споживачів	Прихильність клієнтів. Відмітні властивості товару. Відмітні характеристики товару.	Стратегія диференціації або стратегія спеціалізації

Таблиця 4.17 — Визначення базової стратегії конкурентної поведінки

Чи є проект «першопрохідцем» на ринку	Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?	Чи буде компанія копіювати основні характеристики товару конкурента, які?	Стратегія конкурентної поведінки
Ні, оскільки є товари-замінники, але дані товари-замінники не мають деякого необхідного функціоналу	Так, ціль компанії знайти нових споживачів та, частково, забрати існуючих у конкурентів задля задоволення потреб останніх	Компанія частково копіює характеристики товару конкурента, основна ціль компанії розробка нового унікального функціоналу, з підтримкою основного функціоналу конкурентів	Стратегія заняття конкурентної ніші

Таблиця 4.18 — Визначення стратегії позиціонування

№	Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартап-проекту	Вибір асоціацій, які мають сформувати комплексну позицію власного проекту

Продовження таблиці 4.18

1	Доступна ціна, зрозумілий інтерфейс, зручність у використанні, нові можливості в одному сервісі	Стратегія диференціації	Швидке вирішення поставлених задач, зрозуміло навіть без інструкцій. Легкість і простота у використанні. Доступність через ціну.	стандарти якості; метрики ПЗ;
---	-------------------------------------------------------------------------------------------------------------------	----------------------------	-------------------------------------------------------------------------------------------------------------------------------------------------------	----------------------------------

Відповідно до проведеного аналізу можна зробити висновок, що робота стартап-компанії буде спланована наступним чином: буде виконаний за стратегією диференціації і буде поширюватись товар відмінний за властивостями від своїх аналогів, дотримуючись у конкурентній поведінці стратегії типу «виклику лідера», тобто випускається один товар для усіх можливих споживачів [18].

4.5 Розроблення маркетингової програми стартап-проєкту

Таблиця 4.19 — Визначення ключових переваг концепції потенційного товару

№	Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити)
1	Програмне забезпечення обробки оцифрованих зображень	Редагування, перетворення та відправка зображень на сторонній сервіс	Розрахункові показники, точність та достовірність яких можна оцінювати; простота, кількість вхідних параметрів; самостійність програмної системи.

Таблиця 4.20 — Опис трьох рівнів моделі товару

Рівні товару	Сутність та складові		
1) Товар за задумом	Програмного забезпечення обробки оцифрованих зображень за допомогою хмарних технологій		
2) Товар у реальному виконанні	Властивості/характеристики	М/Нм	Вр/Тх/Тл/Е/Ор
	Ціна	М	Вр
	Наявність всього необхідного інструменту для того щоб редагувати та перетворювати зображення	Нм	Тх/Тл
	Кросплатформеність	Нм	Тх/Тл
	Швидка та надійна робота серверу	М	Вр/Тх/Тл/Е
	Приватність	Нм	Тх/Тл
	Інтеграція з різними продуктами	Нм	Тх/Тл
	Технічна підтримка	М	Вр/Тх/Тл
	Якість: архітектура та програмне забезпечення		
	Марка: НТУУ КПІ ім. Сікорського		
3) Товар із підкріпленням	До продажу: наявна повна документація		
	Після продажу: додаткова підтримка спеціалістів налаштування, підтримка з боку розробника		
За рахунок чого потенційний товар буде захищено від копіювання			

Таблиця 4.21 — Визначення меж встановлення ціни

Рівень цін на товари-замінники	Рівень цін на товари-аналоги	Рівень доходів цільової групи споживачів	Верхня та нижня межі встановлення ціни на товар/послугу
3500 грн	1800 грн	25000 грн	1500 грн

Таблиця 4.22 — Формування системи збуту

Специфіка закупівельної поведінки цільових клієнтів	Функції збуту, які має виконувати постачальник товару	Глибина каналу збуту	Оптимальна система збуту
Бажання отримати більше можливостей за менші гроші	Залучення клієнтської бази, таргетована реклама та продаж	0 – канал нульового рівня адже виробник безпосередньо буде продавати продукт клієнтам [21]	Через сайт самого виробника його компанії

Таблиця 4.23 — Концепція маркетингових комунікацій

№	Специфіка поведінки цільових клієнтів	Канали комунікацій, якими користуються цільові клієнти	Ключові позиції, обрані для позиціонування	Завдання рекламного повідомлення	Концепція рекламного звернення
---	---------------------------------------	--------------------------------------------------------	--------------------------------------------	----------------------------------	--------------------------------

Продовження таблиці 4.23

1	Клієнти дізнаються про товар з відео реклами в інтернеті, а саме інстаграми телеграми соцмережі різні інформаційні виставки та форуми	Інтернет та інформаційні форуми	Низька ціна Легкий і простий у використанні продукт	Ведення інформативної політики про існування нового продукту та його переваг	Витрачай менше часу на редагування зображень, працею швидко та ефективно
---	---------------------------------------------------------------------------------------------------------------------------------------	---------------------------------	-----------------------------------------------------	------------------------------------------------------------------------------	--------------------------------------------------------------------------

Як результат було створено ринкову програму, що включає в себе визначення ключових переваг концепції потенційного товару, опис моделі товару, визначення меж встановлення ціни, формування системи збуту та концепцію маркетингових комунікацій.

4.6 Висновки по розділу 4

В четвертому розділі описано стратегії та підходи з розроблення стартап-проекту, визначено наявність попиту, динаміку та рентабельність роботи ринку, як висновок було вказано що існує можливість ринкової комерціалізації проекту. Розглянувши потенційні групи клієнтів, бар'єри входження, стан конкуренції та конкурентоспроможність проекту було встановлено що проект є перспективним. Розглянуто та обрано альтернативу впровадження стартап-проекту та доведено доцільність подальшої імплементації проекту.

Отже, ринкова (маркетингова) програма орієнтовано має бути побудована таким чином:

- розробка продукту;
- вибір сегменту ринку та пошук клієнтів;
- стратегія розвитку;
- стратегія розподіленості, тобто формування конкурентоспроможності досягається шляхом надання споживачу товару, якого той потребує;
- стратегія конкурентної поведінки – стратегія виклику лідера, тобто на споживчому ринку націлювати на всіх можливих споживачів.

Динаміка ринкового середовища на сьогоднішній день і ще на багато років вперед є і будуть залишатись сприятливими для впровадження описаної системи. Конкурентні переваги створеного продукту очевидні. На вітчизняному ринку аналогів майже не існує, а існуючі – вкрай низької якості. На міжнародному ринку конкуренція наявна та буде рости, якщо не підтримувати та не розвивати свій продукт. Також, після проведення аналізів можливого цільового сегменту споживачів, потреб споживачів та можливого попиту, динаміки ринку та рентабельності роботи на ринку, можна однозначно зробити висновок, що створений проект доцільний до комерціалізації.

ВИСНОВКИ

В даній магістерській дисертації було розглянуто питання пов'язане з проектуванням архітектури програмного забезпечення по обробці оцифрованих зображень з використанням хмарних технологій.

В ході роботи було сформовано та вирішено наступні задачі:

- проведено аналіз архітектурних рішень та систем;
- обрано необхідний функціонал для роботи з зображеннями, як для сторонніх продуктів, які можуть інтегрувати розроблене програмне забезпечення;
- проаналізовано стек технологій, який базується на основі тенденцій розвитку ринку хмарних технологій, а саме: Google Cloud, Amazon Cloud, Microsoft Azure;
- спроектовано хмарну архітектуру по обробці оцифрованих зображень з можливістю легкого масштабування;
- зроблено аналіз отриманих результатів.

Була поставлена та виконана мета, яка полягала в тому, щоб спроектувати сервіс, який надавав би можливість інтеграції з сторонніми продуктами, підвищив швидкість передачі зображень на сторонній сервіс та за допомогою використаних хмарних технологій, зменшив ресурси, які потрібні для підтримки серверу в залежності від навантаження.

В першому розділі було зроблено аналіз та огляд актуальних рішень та підходів, зроблено порівняльну характеристику, виявлено основні недоліки та обрано необхідний функціонал та перспективні можливості програмного забезпечення.

В другому розділі проведено порівняння трьох конкуруючих хмарних корпорацій і було прийнято рішення використовувати AWS. Зроблено аналіз основних сервісів.

В третьому розділі було описано функціональні та нефункціональні можливості системи, спроектовано та описано архітектуру програмного забезпечення. Проведено експериментальні дослідження та надано оцінку ефективності спроектованої системи

і як результат визначено основні переваги в порівнянні з іншими архітектурами. Виявлено, що затримка стає мінімальною за рахунок використання хмарних сервісів, в порівнянні зі звичайною архітектурою MVC, а також збільшення швидкості взаємодії компонентів на 7%.

Для розробки описаного архітектурного рішення системи було використано наступні технології: Auth0, Stripe, Amazon Route 53, AWS WAF, Amazon CloudFront, Amazon S3, Amazon API Gateway, AWS Lambda, Amazon DynamoDB, Amazon SQS. Всі вище перераховані інструменти, дозволяють створити швидку та надійну хмарну архітектуру, для роботи з оцифрованими зображеннями.

Наукова новизна даної магістерської дисертації полягає в наступному:

удосконалено:

- архітектурне рішення програмного забезпечення системи по обробці зображень з можливістю відправки на сторонній сервіс;
- забезпечено горизонтального та вертикального масштабування описаної архітектури за рахунок використання хмарних технологій;
- зменшення ресурсів для підтримки роботоспроможності серверу;
- збільшення швидкості відправки зображень на сторонній сервіс.

Результати були опубліковано у статті.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

- 1) Colorize B&W Photos URL: <https://photomyne.com/colorize-app> (дата звернення 1.09.2021).
- 2) Pho. URL: <https://pho.to/> (дата звернення 1.09.2021).
- 3) PhotoshopOnline URL: <https://photoshoponline> (дата звернення 15.09.2021).
- 4) AWS or Azure or GCP URL: <https://allinfo.tech/services/aws-azure-gcp> (дата звернення 18.11.2021).
- 5) AWS vs Azure vs Google Cloud URL: <https://dinarys.com/ru/blog/comparison-of-aws-vs-azure-vs-google-cloud> (дата звернення 8.09.2021).
- 6) R. S. Sutton, “Two problems with backpropagation and other steepest-descent learning procedures for networks”, in Proceedings of the Eighth Annual Conference of the Cognitive Science Society, Hillsdale, NJ: Erlbaum, 1986.
- 7) Google Cloud URL: <https://cloud.google.com/free/docs/aws-azure-gcp-service-comparison> (дата звернення 27.09.2021).
- 8) C. Doersch, J. Donahue, and T. Darrell, “Data-dependent initializations of convolutional neural networks”, CoRR, vol. abs/1511.06856, 2015.
- 9) Comparing the Big 3: AWS vs Azure vs GCP URL: <https://www.n-ix.com/comparing-big-3-aws-azure-gcp/> (дата звернення 26.10.2021).
- 10) AWS vs Azure vs Google Cloud. What's the best cloud platform for enterprise? URL: <https://www.computerworld.com/article/3429365/aws-vs-azure-vs-google-whats-the-best-cloud-platform-for-enterprise> (дата звернення 28.10.2021).
- 11) AWS vs. Azure vs. Google Cloud: Which Is Better? URL: <https://sam-solutions.us/aws-vs-azure-v-s-google-cloud-which-is-better/> (дата звернення 24.10.2021).
- 12) Difference between AWS, Azure, and Google Cloud Platform URL: <https://www.javatpoint.com/aws-vs-azure-vs-google-cloud-platform> (дата звернення 24.10.2021).

- 13) AWS URL: <https://aws.amazon.com/ru/> (дата звернення 12.10.2021).
- 14) Azure vs Google Cloud URL: <https://azure.microsoft.com/ru-ru/> (дата звернення 12.10.2021).
- 15) AWS vs Google Cloud URL: <https://www.educba.com/google-cloud-vs-aws/> (дата звернення 20.10.2021).
- 16) Царегородцев А.В., Качко А.К. Забезпечення інформаційної безпеки на хмарній архітектурі організації // Національна безпека. 2011. № 5. С. 25–34.
- 17) Телипенко Е.В. Система підтримки прийняття рішень під час управління ризиком банкрутства підприємства: автореф. дисс. канд. тех. наук: 05.13.10 – Новосибірськ, 2013. – 24 с.
- 18) Захарова А.А. Система підтримки прийняття рішень щодо стратегії інноваційного розвитку регіону: монографія / А.А. Захарова; Юргинський технологічний інститут. – Томськ: Вид-во Томського політехнічного університету, 2011. – 144 с.
- 19) Comparing Kubernetes Services on AWS vs. Azure vs. GCP URL: <https://www.sumologic.com/blog/kubernetes-aws-azure-gcp/> (дата звернення 25.10.2021).
- 20) Тест хостингу виділеного сервера – порівняння 2021 URL: <https://test-vergleiche.com/uk/> (дата звернення 29.10.2021).
- 21) Розроблення стартап-проекту. Методичні рекомендації до виконання розділу магістерських дисертацій для студентів інженерних спеціальностей / За заг. ред. О.А. Гавриша. – Київ : НТУУ «КПІ», 2016. – 28 с.
- 22) Newell, A.; Yang, K.; Deng, J. Stacked hourglass networks for human pose estimation. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 8–16 October 2016; pp. 483–499.
- 23) What is DynamoDB? URL: <https://www.dynamodbguide.com/what-is-dynamo-db/> (дата звернення 1.10.2021).

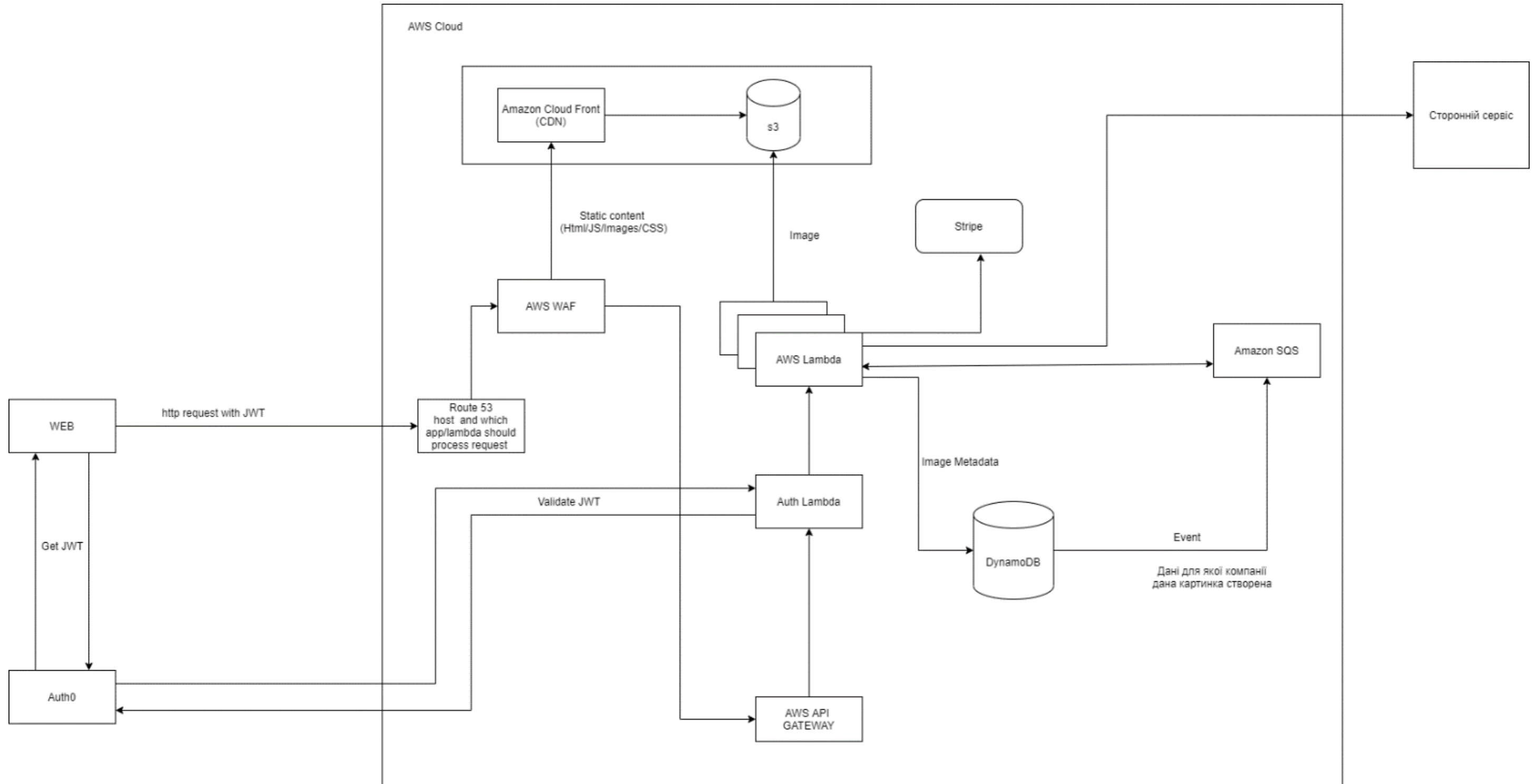
- 24) JWT.IO allows you to decode, verify and generate JWT. URL: <https://jwt.io/> (дата звернення 5.10.2021).
- 25) AWS PrivateLink URL: <https://aws.amazon.com/privatelink/?privatelink-blogs> (дата звернення 1.10.2021).
- 26) S3 Background URL: <https://darkbit.io/blog/understanding-s3-block-public-access> (дата звернення 26.11.2021).
- 27) Amazon Simple Queue Service (SQS) URL: <https://aws.amazon.com/sqs/> (дата звернення 29.11.2021).
- 28) Swagger API Development for Everyone URL: <https://swagger.io/> (дата звернення 20.11.2021).
- 29) Amazon CloudFront URL: <https://aws.amazon.com/ru/cloudfront/> (дата звернення 11.11.2021).
- 30) AWS WAF – Брандмауер для інтернет-додатків URL: <https://aws.amazon.com/ru/waf/> (дата звернення 17.11.2021).
- 31) What are AWS WAF, AWS Shield, and AWS Firewall Manager URL: <https://docs.aws.amazon.com/waf/latest/developerguide/> (дата звернення 10.11.2021).
- 32) Amazon Route 53 URL: <https://aws.amazon.com/ru/route53/> (дата звернення 5.11.2021).
- 33) What is Amazon Route 53? URL: <https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/Welcome.html> (дата звернення 28.11.2021).
- 34) Amazon Route 53 Documentation URL: <https://docs.aws.amazon.com/route53/index.html> (дата звернення 19.11.2021).
- 35) Stripe - payments infrastructure for the internet URL: <https://stripe.com/> (дата звернення 14.11.2021).

- 36) D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Parallel distributed processing: Explorations in the microstructure of cognition, vol. 1”, in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, Vol. 1, D. E. Rumelhart, J. L. McClelland, and C. PDP Research Group, Eds., Cambridge, MA, USA: MIT Press, 1986, ch. Learning Internal Representations by Error Propagation, pp. 318–362, isbn: 0-262-68053-X.
- 37) S. Iizuka, E. Simo-Serra, and H. Ishikawa, “Let there be color!: Joint end-to-end learning of global and local image priors for automatic image colorization with simultaneous classification”, *ACM Trans. Graph.*, vol. 35, no. 4, 110:1–110:11, Jul. 2016, issn: 0730-0301.
- 38) Y. Lecun, Y. Bengio, and G. Hinton, “Deep learning”, *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015, issn: 0028-0836.
- 39) N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A simple way to prevent neural networks from overfitting”, *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014.
- 40) Пояснення та застосування CNN URL:
<https://evergreens.com.ua/ua/articles/cnn.html> (дата звернення 21.11.2021).

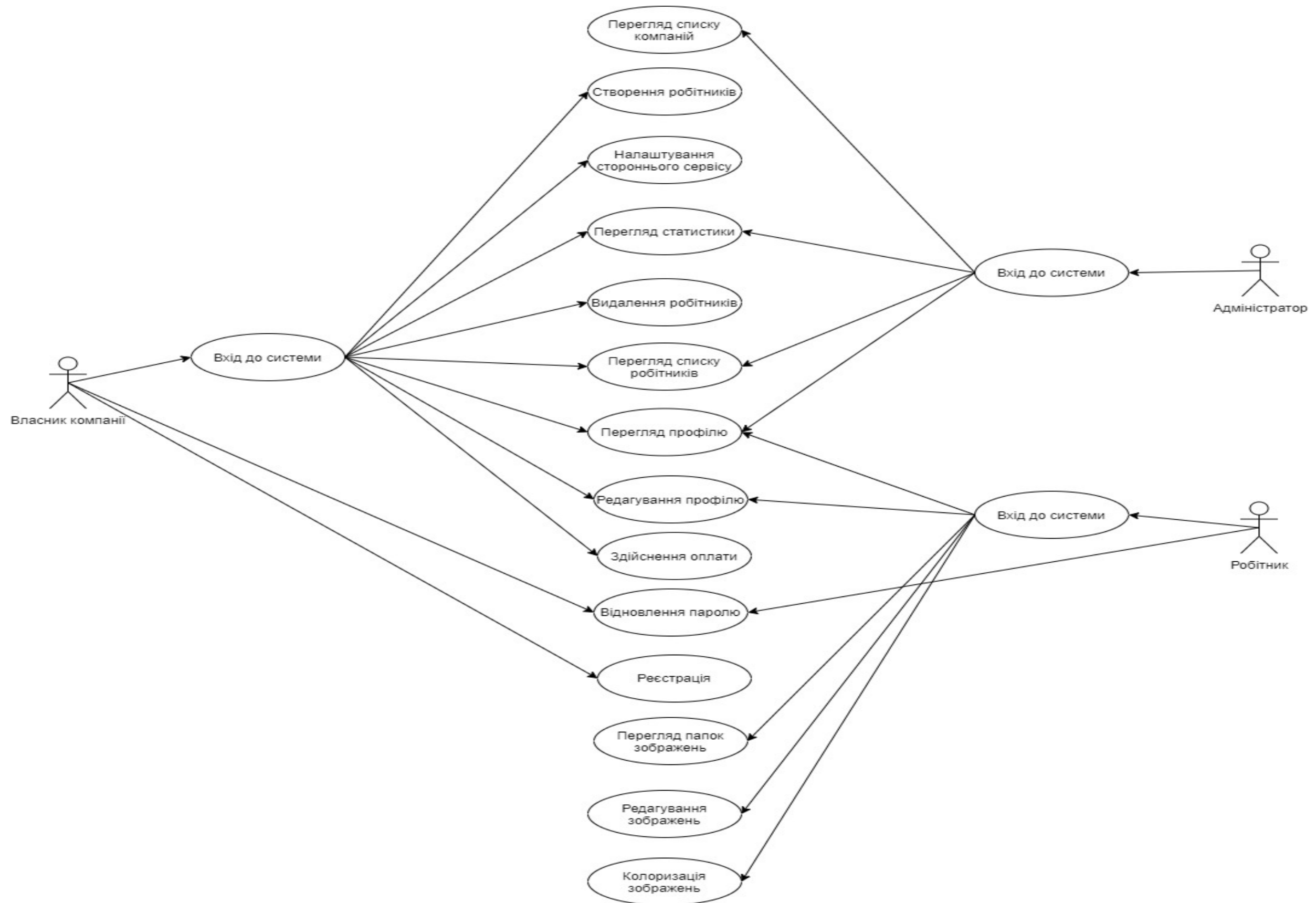
ДОДАТКИ

ДОДАТОК А

Схема архітектури програмного забезпечення



ДОДАТОК Б
Діаграма використання



ДОДАТОК В

Алгоритм обробки зображень на стронній сервіс



ДОДАТОК Г

Результати перевірки роботи на співпадіння

```
Admin.tsx
import { faEllipsisV, faPlus } from '@fortawesome/free-solid-svg-icons';
import { FontAwesomeIcon as Icon } from '@fortawesome/react-fontawesome';
import React from 'react';
import Button from '../components/Button';
import FlatList from '../components/FlatList';
import Title from '../components/Title';
import Image from '../components/Image';
import { useFetch } from '../hooks/useFetch';
import './admin.scss';

const Admin: React.FC = () => {
  const { data } = useFetch(
    'https://api.unsplash.com/photos/?client_id=G36HLpMpORyLoY45oF9Vqy1GjIrGA3zI22zvP
OOcnzQ'
  );

  return (
    <>
      <Button
        variant="primary"
        text={
          <>
            <Icon className="icon" icon={faPlus} />
            Add image
          </>
        }
      />
      <Title variant="primary">Images</Title>
      <FlatList
        data={data}
        className="user-image-list"
        itemProps={{ className: 'user-image-item' }}
        component={({ urls, description, user }) => (
          <div className="upload-image-wrapper">
            <Image className="upload-image" src={urls.small} />

            <div className="upload-image-info">
              <h4 className="upload-image-description">
                {description || 'Image'}
              </h4>
              <h4 className="upload-image-name">{user.name}</h4>
            </div>

            <Button
              className="upload-image-btn"
              text={<Icon icon={faEllipsisV} />}
            />
          </div>
        )}
      />
    </>
  );
};
```

```

    </>
  );
};

export default Admin;
AdminHome.tsx
import {
  faImages,
  faInfo,
  faServer,
  faUser,
  faUserTie,
} from '@fortawesome/free-solid-svg-icons';
import React from 'react';
import { Stat, StatBox } from '../components/Stat';
import Title from '../components/Title';

const AdminHome: React.FC = () => {
  return (
    <>
      <Title variant="primary">Statistics</Title>

      <StatBox column="2">
        <Stat title="Unique user" value="50" icon={faUserTie} variant="dark" />
        <Stat
          title="Images converted"
          value="124"
          icon={faInfo}
          variant="primary"
        />
      </StatBox>

      <Title variant="secondary">Weekly</Title>

      <StatBox column="3">
        <Stat
          title="User registrated"
          value="4"
          icon={faUserTie}
          variant="secondary"
        />
        <Stat
          title="Images converted"
          value="22"
          icon={faImages}
          variant="info"
        />
        <Stat

```

```

        title="Site visited"
        value="124"
        icon={faUser}
        variant="warning"
      />
    </StatBox>

    <Title variant="secondary">Server</Title>

    <StatBox column="1">
      <Stat
        title="Memory used"
        value="23.4MB"
        icon={faServer}
        variant="purple"
      />
    </StatBox>
  </>
);
};

export default AdminHome;
Home.tsx
import React from 'react';
import { Link } from 'react-router-dom';
import image from '../../assets/images/photo.svg';
import Button from '../../components/Button';
import Image from '../../components/Image';
import Title from '../../components/Title';
import './home.scss';

const Home: React.FC = () => {
  return (
    <div className="container">
      <section className="home-section">
        <div className="block">
          <Title variant="primary">
            You can edit your photo and send it to your service
          </Title>
          <Link className="link" to="/sign-up">
            <Button variant="primary">Get Started</Button>
          </Link>
        </div>
        <Image className="home-image" src={image} />
      </section>
    </div>
  );
};

export default Home;

Settings.tsx
import { Form, Formik } from 'formik';
import React from 'react';
import Button from '../../components/Button';
import { Checkbox, Input } from '../../components/Form';
import Title from '../../components/Title';

```

```

import './settings.scss';

const Settings: React.FC = () => {
  return (
    <>
      <Title variant="primary" tag="h2">
        Settings
      </Title>
      <Formik
        initialValues={{ token: '', email: '' }}
        onSubmit={(_, action) => action.resetForm()}
      >
        {({ resetForm }) => (
          <Form className="form settings-form card">
            <Title variant="secondary" tag="h3">
              Take your choose
            </Title>
            <Input variant="primary" name="token" placeholder="Access token" />
            <Input
              variant="primary"
              name="token"
              placeholder="Arhive destination"
            />
            <Input
              variant="primary"
              name="token"
              placeholder="Folder destination"
            />
            <Checkbox name="email" placeholder="Get messages on my email" />
            <div className="row">
              <Button type="submit" variant="primary" text="Save" />
              <Button
                type="button"
                variant="danger"
                text="Clear"
                outline
                onClick={resetForm}
              />
            </div>
          </Form>
        )}
      </Formik>
    </>
  );
};

```

```
export default Settings;
```

```
SignIn.tsx
```

```

import React from 'react';
import { Form, Formik } from 'formik';
import * as yup from 'yup';
import { Input } from '../components/Form';
import Button from '../components/Button';
import Notification from '../components/Notification';
import { useToast } from '../hooks/useToast';
import { useNavigate } from 'react-router-dom';

```

```

import './signIn.scss';
import Title from '../..//components/Title';

const SignIn: React.FC = () => {
  const { push } = useToast();
  const navigate = useNavigate();

  return (
    <Formik
      initialValues={{
        email: '',
        password: '',
      }}
      validationSchema={yup.object({
        email: yup.string().email().required(),
        password: yup.string().min(8).required(),
      })}
      onSubmit={(_, action) => {
        push((id) => (
          <Notification
            type="success"
            text="User successfully created"
            id={id}
          />
        ));
        navigate('/admin');
        action.resetForm();
      }}
    >
    {() => (
      <div className="sign-form-outer sign-in-wrapper">
        <Form className="form sign-form card">
          <Title variant="secondary">Sign in</Title>
          <Input placeholder="Email" variant="primary" name="email" />
          <Input
            placeholder="Password"
            type="password"
            variant="primary"
            name="password"
          />
          <Button type="submit" outline variant="primary" text="Sign in" />
        </Form>
      </div>
    )}
  </Formik>
);
};

```

```
export default SignIn;
```

```
SignUp.tsx
```

```

import React from 'react';
import { Form, Formik } from 'formik';
import * as yup from 'yup';
import { Input } from '../..//components/Form';
import Button from '../..//components/Button';
import Notification from '../..//components/Notification';
import { useToast } from '../..//hooks/useToast';

```

```

import { Link, useNavigate } from 'react-router-dom';
import Title from '../..components/Title';
import Text from '../..components/Text';
import './signUp.scss';

const SignUp: React.FC = () => {
  const { push } = useToast();
  const navigate = useNavigate();

  return (
    <Formik
      initialValues={{
        email: '',
        userName: '',
        password: '',
      }}
      validationSchema={yup.object({
        email: yup.string().email().required(),
        userName: yup.string().required(),
        password: yup.string().min(8).required(),
      })}
      onSubmit={(_, action) => {
        push((id) => (
          <Notification
            type="success"
            text="User successfully created"
            id={id}
          />
        ));
        navigate('/admin');
        action.resetForm();
      }}
    >
    {() => (
      <div className="sign-form-outer sign-up-wrapper">
        <Form className="form sign-form card">
          <Title variant="secondary">Sign up</Title>
          <Input placeholder="Email" variant="primary" name="email" />
          <Input placeholder="User name" variant="primary" name="userName" />
          <Input
            placeholder="Password"
            type="password"
            variant="primary"
            name="password"
          />
          <Button type="submit" variant="primary" text="Sign up" />

          <Text>
            Already have account? &nbsp;
            <Link className="link" to="/sign-in">
              Sign in
            </Link>
          </Text>
        </Form>
      </div>
    )}
  </Formik>
);

```

```

};

export default SignUp;

User.tsx
import { faEllipsisV } from '@fortawesome/free-solid-svg-icons';
import { FontAwesomeIcon as Icon } from '@fortawesome/react-fontawesome';
import React from 'react';
import Button from '../../components/Button';
import classNames from '../../utils/classNames';
import initials from '../../utils/initials';
import { randomIndex } from '../../utils/randomIndex';
import './user.scss';

interface IUser {
  name: string;
  email: string;
}

const variants = [
  'primary',
  'secondary',
  'dark',
  'danger',
  'warning',
  'info',
  'purple',
];

const User: React.FC<IUser> = ({ name, email }) => {
  return (
    <div className="user">
      <div
        className={classNames(
          'user-initials',
          `user--${variants[randomIndex(variants.length)]}`
        )}
      >
        {initials(name)}
      </div>

      <div className="user-info">
        <h4 className="user-name">{name}</h4>

        <h4 className="user-email">{email}</h4>
      </div>

      <Button className="user-btn" text={<Icon icon={faEllipsisV} />} />
    </div>
  );
};

export default User;

serverless.yml
service: Digitized image processing

provider:

```

```

name: aws
profile: default
iamRoleStatements:
  - Effect: 'Allow'
    Action:
      - 'sqs:SendMessage'
      - "sqs:GetQueueUrl"
      - "sqs:ReceiveMessage"
      - "sqs:ListQueues"
    Resource: 'arn:aws:sqs:${self:provider.region}:*'
  - Effect: 'Allow'
    Action:
      - 'dynamodb:Query'
      - 'dynamodb:Scan'
      - 'dynamodb:GetItem'
      - 'dynamodb:BatchGetItem'
      - 'dynamodb:BatchWriteItem'
      - 'dynamodb:PutItem'
      - 'dynamodb:UpdateItem'
      - 'dynamodb>DeleteItem'
      - 'dynamodb:DescribeTable'
      - 'dynamodb:CreateTable'
    Resource: "arn:aws:dynamodb:${opt:region, self:provider.region}:*:*"
  - Effect: 'Allow'
    Action:
      - 'lambda:InvokeFunction'
    Resource:
      - Fn::Join:
          - ':'
          - - arn:aws:lambda
            - Ref: AWS::Region
            - Ref: AWS::AccountId
            - function:${self:service}-${opt:stage, self:provider.stage}-*
  - Effect: 'Allow'
    Action:
      - 'execute-api:*'
    Resource: "arn:aws:execute-api:*:*:*"
dynamo_hosts:
  local-dev: http://localhost:8000
  dev: https://dynamodb.eu-central-1.amazonaws.com
  stage: https://dynamodb.eu-central-1.amazonaws.com
  prod: https://dynamodb.eu-central-1.amazonaws.com

environment:
  COMPANY_TABLE: companies-${opt:stage, self:provider.stage}
  IMAGES_TABLE: images-${opt:stage, self:provider.stage}
  SEND_IMAGES_QUEUE_NAME: { Ref: SendImagesToCustomerService }

  # ${opt:stage, self:provider.stage} - use {provider.} for dev, stage, prod

package:
  exclude:
    - node_modules/**
    - venv/**

authorizerParams: &requestAuthorizer
  name: RequestAuthorizer

```

```
resultTtlInSeconds: 0
type: request
arn:
  'Fn::ImportValue': 'AuthArn-${self:custom.stage}'
```

functions:

```
get_image_by_id:
  handler: apps/image/handlers.get_image_by_id
  events:
    - http:
        path: v1/images/{id}
        method: get
        authorizer: *requestAuthorizer
```

```
get_images_by_folder_id:
  handler: apps/image/handlers.get_images_by_company_id
  events:
    - http:
        path: v1/images/{folder_id}/
        method: get
        authorizer: *requestAuthorizer
```

```
save_images:
  handler: apps/image/handlers.save_images
  events:
    - http:
        path: v1/images/
        method: post
        authorizer: *requestAuthorizer
```

```
change_image:
  handler: apps/image/handlers.change_image
  events:
    - http:
        path: v1/images/{id}
        method: get
        authorizer: *requestAuthorizer
```

```
delete_image:
  handler: apps/image/handlers.delete_image
  events:
    - http:
        path: v1/images/{id}
        method: delete
        authorizer: *requestAuthorizer
```

```
create_folder:
  handler: apps/image/handlers.create_folder
  events:
    - http:
        path: v1/images/folder
        method: post
        authorizer: *requestAuthorizer
```

```
get_list_folders:
  handler: apps/image/handlers.get_folders
  events:
```

```
- http:
  path: v1/images/folder
  method: get
  authorizer: *requestAuthorizer

create_company:
  handler: apps/company/handlers.create
  events:
    - http:
      path: v1/companies
      method: post
      authorizer: *requestAuthorizer

get_company:
  handler: apps/company/handlers.get
  events:
    - http:
      path: v1/companies/{id}
      method: get
      authorizer: *requestAuthorizer

update_company:
  handler: apps/company/handlers.update
  events:
    - http:
      path: v1/companies/{id}
      method: put
      authorizer: *requestAuthorizer

delete_company:
  handler: apps/company/handlers.delete
  events:
    - http:
      path: v1/companies/{id}
      method: delete
      authorizer: *requestAuthorizer

create_user:
  handler: apps/user/handlers.create
  events:
    - http:
      path: v1/users/
      method: post
      authorizer: *requestAuthorizer

update_user:
  handler: apps/user/handlers.update
  events:
    - http:
      path: v1/users/{id}
      method: put
      authorizer: *requestAuthorizer

get_list_users:
  handler: apps/user/handlers.list
  events:
    - http:
      path: v1/users
```

```

        method: put
        authorizer: *requestAuthorizer

delete_user:
  handler: apps/user/handlers.delete
  events:
    - http:
        path: v1/users/{id}
        method: delete
        authorizer: *requestAuthorizer

send_images_to_customer_service:
  handler: apps/sqs.send_images_to_customer_service
  timeout: 300
  events:
    - sqs:
        arn:
            Fn::GetAtt:
                - SendImagesToCustomerService
                - Arn
        batchSize: 1

resources:
  Resources:
    SendImagesToCustomerService:
      Type: "AWS::SQS::Queue"
      Properties:
        QueueName: SendImagesToCustomerService-${opt:stage, self:provider.stage}
        VisibilityTimeout: 300

    CompanyTable:
      Type: 'AWS::DynamoDB::Table'
      DeletionPolicy: Retain
      Properties:
        StreamSpecification:
          StreamViewType: COMPANY
        AttributeDefinitions:
          - AttributeName: id
            AttributeType: S
        KeySchema:
          - AttributeName: id
            KeyType: HASH
        BillingMode: PAY_PER_REQUEST
        TableName: ${self:provider.environment.COMPANY_TABLE}

    ImagesTable:
      Type: 'AWS::DynamoDB::Table'
      DeletionPolicy: Retain
      Properties:
        StreamSpecification:
          StreamViewType: NEW_IMAGES
        AttributeDefinitions:
          - AttributeName: id
            AttributeType: S
        KeySchema:
          - AttributeName: id

```

```
        KeyType: HASH
        BillingMode: PAY_PER_REQUEST
        TableName: ${self:provider.environment.IMAGES_TABLE}
ImagesTable:
  Type: 'AWS::DynamoDB::Table'
  DeletionPolicy: Retain
  Properties:
    StreamSpecification:
      StreamViewType: USER
    AttributeDefinitions:
      - AttributeName: id
        AttributeType: S
    KeySchema:
      - AttributeName: id
        KeyType: HASH
    BillingMode: PAY_PER_REQUEST
    TableName: ${self:provider.environment.USER}

plugins:
  - serverless-requirements
  - serverless-plugin-warmup
  - serverless-offline

custom:
  #warm up lambdas
  warmup:
    enabled: true
    prewarm: true

  stage: ${opt:stage, self:provider.stage}

  serverless-offline:
    httpPort: 3001
    websocketPort: 4001
    hideStackTraces: false

  defaultRegion: 'eu-central-1'
  region: ${opt:region, self:custom.defaultRegion}
  account_id: ${self:custom.stageVars.${self:custom.stage}.account_id}
```

ДОДАТОК Д
Результати перевірки роботи на співпадіння



Имя пользователя:
Лісовиченко Олег Іванович

ID проверки:
1009387026

Дата проверки:
28.11.2021 15:44:33 EET

Тип проверки:
Doc vs Internet + Library

Дата отчета:
28.11.2021 15:51:48 EET

ID пользователя:
76913

Название файла: IT-04мп_Царук_ПЗ

Количество страниц: 43 Количество слов: 8688 Количество символов: 64181 Размер файла: 118.61 KB ID файла: 1009406834

2.38% Совпадения

Наибольшее совпадение: 0.54% с источником из Библиотеки (ID файла: 1000016538)

0.74% Источники из Интернета

3

Страница 45

2.05% Источники из Библиотеки

19

Страница 45

0% Цитат

Исключение цитат выключено

Исключение списка библиографических ссылок выключено

0% Исключений

Нет исключенных источников

Модификации

Обнаружены модификации текста. Подробная информация доступна в онлайн-отчете.

Замененные символы

5