Co zostało do zrobienia?

1. Sprawozdanie
2. Diagram architektury
3. Opis użytych technologii/opis szczegółów technicznych(opisy użytych importów – przede wszystkim ‘boto’ użyte do rekognition i transkrypcji, frontend wykorzystuje chakra UI

Link do yt:

https://youtu.be/dqCGewXc-j0

**Rozpoznanie obrazu z wykorzystaniem rozwiązań chmurowych AWS**

Wykonali:

Julian Przybysz 299213

Łukasz Strep

Jakub Słowikowski

Konrad Dudek

Wstęp:

W niniejszym filmiku omówię wykonany projekt czyli „rozpoznanie obrazu z wykorzystaniem rozwiązań chmurowych AWS”. Projekt wykonano jako aplikację webową, z użyciem pythona, postgreSQL, FastAPI oraz reacta. Aplikacja posiada dwie główne funkcjonalności analizę zdjęcia osoby oraz wykonuje transkrypcję z załączonego zdjęcia. Upload zdjęcia odbywa się do Storege’a S3, następnie zdjęcie jest poddawane analizie dla pierwszej funkcjonalności z wykorzystaniem technologii deep-learning Amazon Rekognition, natomiast druga funkcjonalność wykorzystuje Amazon Textract – serwis który automatycznie wykrywa i ekstrahuje tekst ze zeskanowanego zdjęcia – wykonane w zakładce Image transcription.

Zastosowano 6 technologii:

* Storage S3
* MFA
* Cloudtrail
* Budget
* Amazon rekognition
* Amazontextract

Poniżej użyte technologie:

1. Stworzenie bucketu S3:

Obraz zawierający tekst

Opis wygenerowany automatycznie

Obraz zawierający tekst

Opis wygenerowany automatycznie

Otrzymany bucket:

Obraz zawierający tekst

Opis wygenerowany automatycznie

1. MFA

<https://docs.aws.amazon.com/IAM/latest/UserGuide/id_credentials_mfa_enable.html>

Obraz zawierający tekst

Opis wygenerowany automatycznie

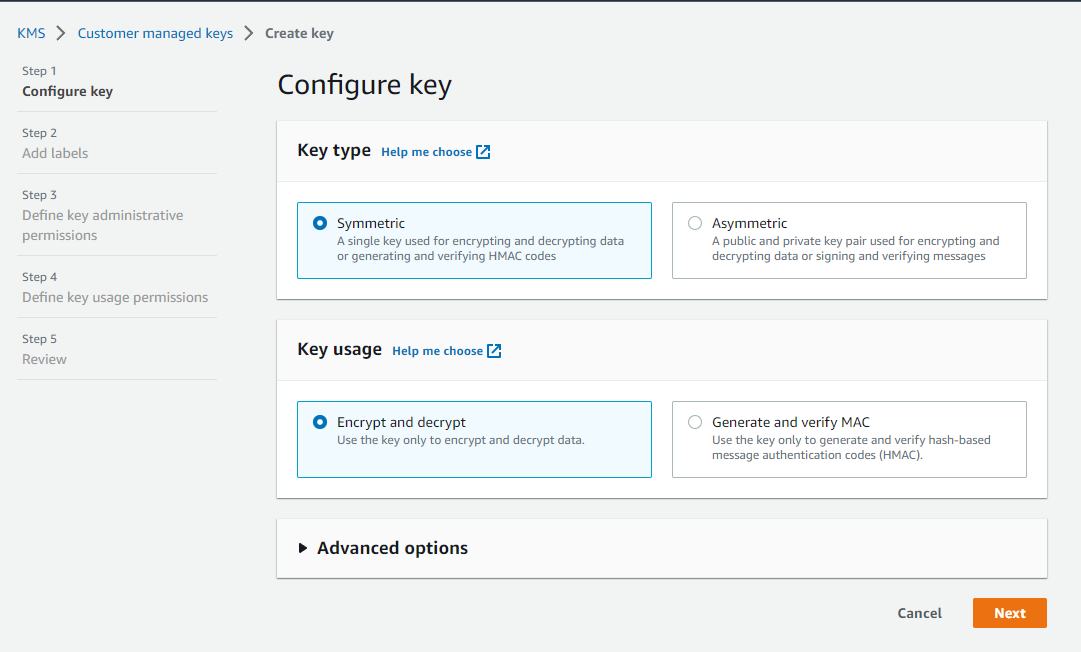
Obraz zawierający tekst

Opis wygenerowany automatycznie

1. Cloud trial

<https://docs.aws.amazon.com/awscloudtrail/latest/userguide/cloudtrail-user-guide.html>

Stworzono potrzebny klucz KMS



Obraz zawierający tekst

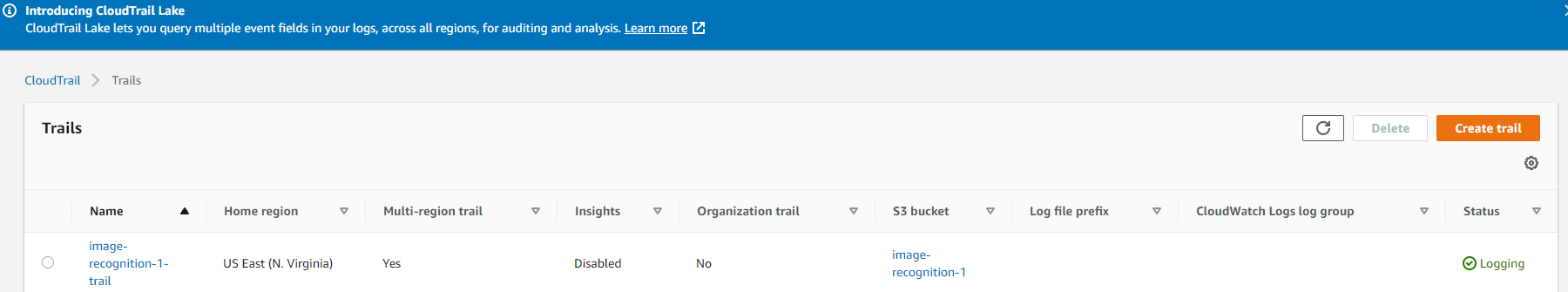
Opis wygenerowany automatycznie

Obraz zawierający tekst

Opis wygenerowany automatycznie

Obraz zawierający tekst

Opis wygenerowany automatycznie



Obraz zawierający tekst

Opis wygenerowany automatycznie

1. Budget

<https://docs.aws.amazon.com/cost-management/latest/userguide/budgets-managing-costs.html>

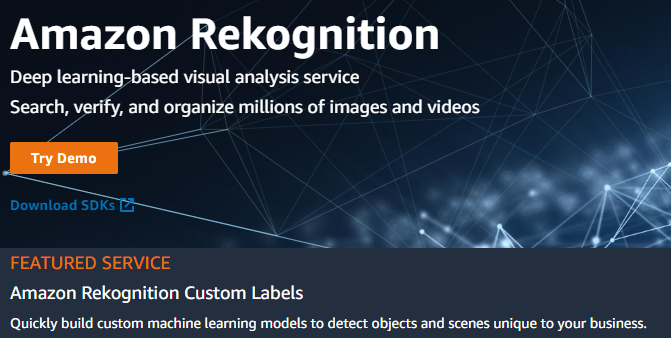
Obraz zawierający tekst

Opis wygenerowany automatycznie

Obraz zawierający tekst

Opis wygenerowany automatycznie

1. Amazon rekognition



Obraz zawierający tekst

Opis wygenerowany automatycznie

1. Amazon textract:

Obraz zawierający tekst

Opis wygenerowany automatycznie

Obraz zawierający tekst

Opis wygenerowany automatycznie

**Budowa:**

Dokeryzacja, użycie postgre SQL

Obraz zawierający tekst

Opis wygenerowany automatycznie

**Fastapi:**



Inicjalizacja + CORS middleware:

Obraz zawierający tekst

Opis wygenerowany automatycznie

Running uvicorn:



Obraz zawierający tekst

Opis wygenerowany automatycznie

Uruchomienie frontendu z npm start

Obraz zawierający tekst

Opis wygenerowany automatycznie

Obraz zawierający tekst

Opis wygenerowany automatycznie

Adminer do PostgreSQL:



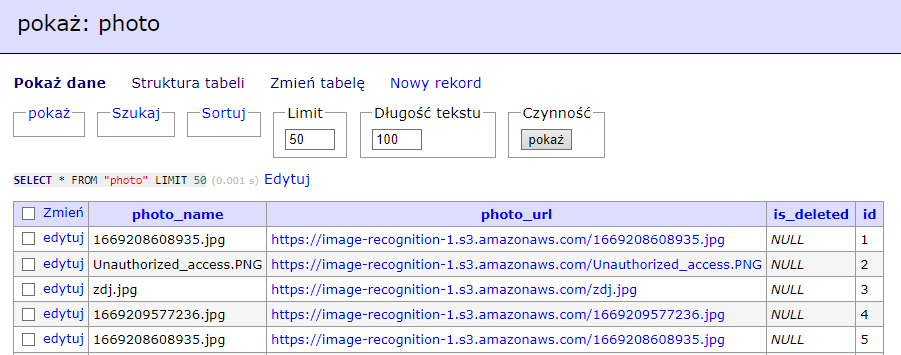
Stworzenie dwóch tabel, do przechowywania nazw uploadowanch obrazów oraz linków do nich w storege’u S3, są stworzone dwie tabele – jedna do analizy twarzy/osoba a druga do trankrypcji:

Obraz zawierający stół

Opis wygenerowany automatycznie



Przykład co siedzi w środku:



Obraz zawierający tekst

Opis wygenerowany automatycznie

Głównym plikiem jest server.py:

Użyte importy

Obraz zawierający tekst

Opis wygenerowany automatycznie

Funcjonalność 1(analiza obrazu) krok po kroku

Na frontendzie wybierany jest plik i po naciśnięciu przycisku upload

Obraz zawierający tekst

Opis wygenerowany automatycznie

wywoływany jest endpoint z fastapi POST /photos: *(#analiza obrazu jest w frontend->src->pages->pricing, a transkrypcja w pages -> about)*

Obraz zawierający tekst

Opis wygenerowany automatycznie

Wtedy endpoint łączy się z amazonem poprzez boto 3 i wgrywa na s3 obraz na odpowiedni bucket, pobiera url, łączy się z bazą danych i dodaje w bazie nowy rekord do tabeli photos z nazwą pliku i url do bucketu s3.

Obraz zawierający tekst

Opis wygenerowany automatycznie

Następnie frontend wywołuje endpoint GET, z którego wyciąga plik json z nazwą zdjęcia, url do S3 oraz opisem(description):

Obraz zawierający tekst

Opis wygenerowany automatycznie

Następnie endpoint łączy się z bazą, bierze pierwszy od góry rekord z tabeli ‘photo’

Obraz zawierający tekst

Opis wygenerowany automatycznie

następnie wywołuje funkcję ‘get\_image\_description’ w pliku orm, która łączy się z technologią AWS rekognition:

Obraz zawierający tekst

Opis wygenerowany automatycznie

Z tej funcji wyciągany jest string zawierający informację o płci, zakresie wieku, uśmiechu oraz emocji.

Następnie w endpoincie informacje z bazy i funkcji rekognition są przekazywane do photo\_model, który następnie zwracany jest do frontendu, który wyświetla obraz i opis:

Obraz zawierający tekst

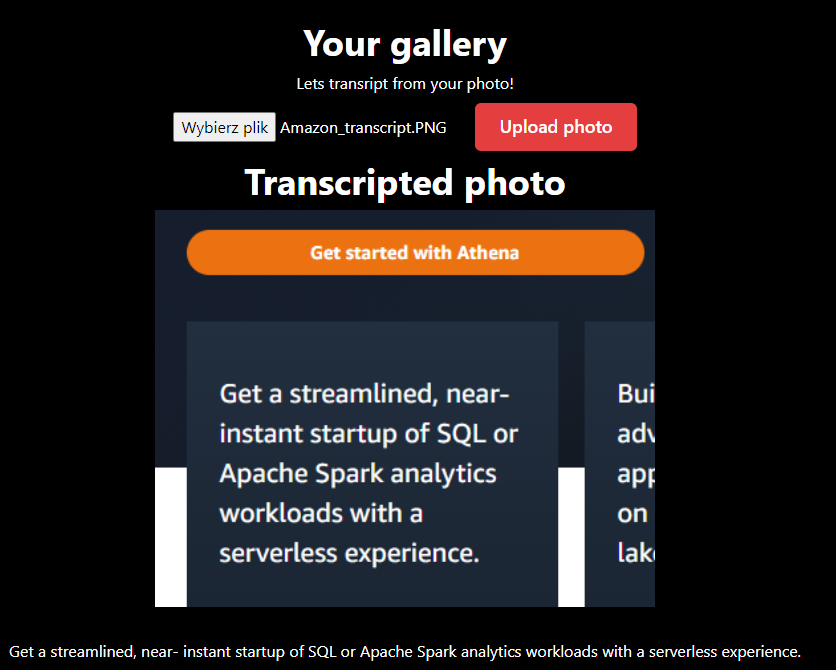
Opis wygenerowany automatycznie

Obraz zawierający tekst, osoba, wewnątrz, uśmiech

Opis wygenerowany automatycznie

**Druga funkcjonalność:**

Budowa jest analogiczna tylko korzystamy z innej tabeli postgres – transcribe oraz innego bucketu i wywołujemy inną funkcję get\_image\_transcription, odwołującą się do innej technologii – AWS textract.



#warto opisać może coś z frontendu jeszcze

Obraz zawierający tekst, zrzut ekranu, monitor, ekran

Opis wygenerowany automatycznie

Zakładki:

