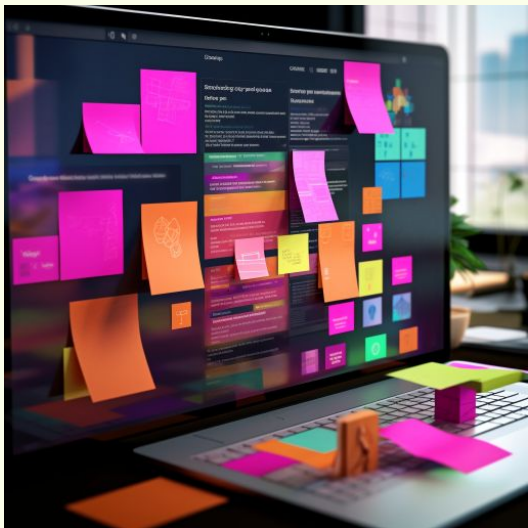


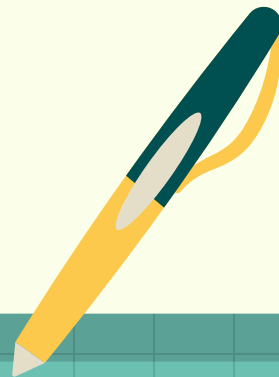
Erasing Digital Sticky Notes: A Software Developer's Adventure in PII Data Removal



Daniel Aniszkiewicz

1st June 2023

AWS Community Day Warsaw



O mnie



Daniel Aniszkiewicz
Serverless enthusiast

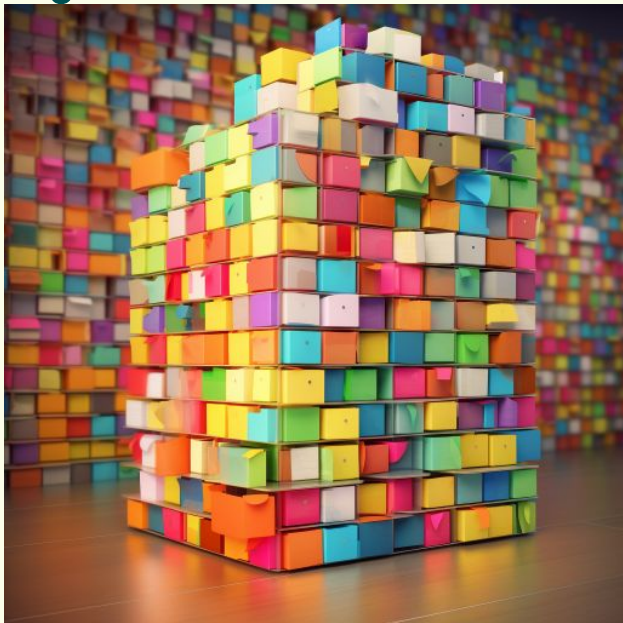
Senior Software Engineer - obecnie Algoteque
AWS Community Builder (Serverless) Q4 2021
Co-organizer AWS User Group Wrocław

Case

O firmie

- firma ubezpieczeniowa.
- sprzedaje polisy ubezpieczeniowe.
- system CRM gdzie są wszystkie dane klientów.

Architektura



***Monolit (Ruby on Rails) z bazą
Postgres na AWS.***

○ Stare czasy



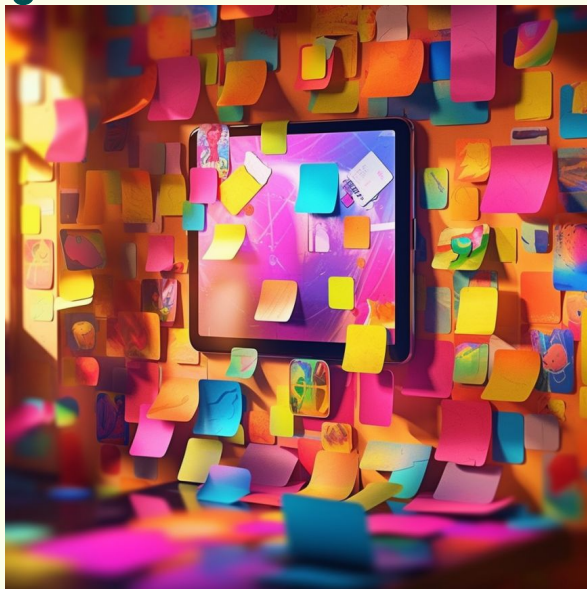
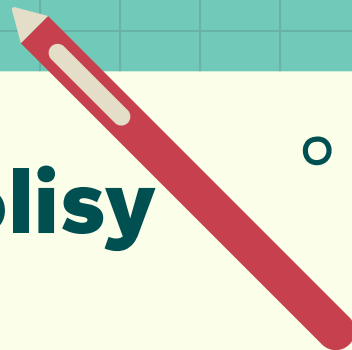
Problem

Do każdej polisy sprzedawca może dodać ogólnodostępne notatki dla pracowników całej firmy.

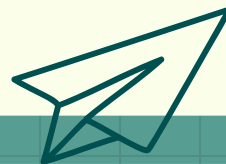
Ktoś zauważył, że w notatkach są dane kart kredytowych i nie tylko.



Przykład notatki do polisy



"Please charge this client for the renewal of the insurance policy. His credit card number is xxx, pin xxx, cvc xxx. His SSN is xxx and his full residential address is xxx."



Houston, mamy problem!



- 500 tysięcy notatek, potencjalnie każda może mieć w sobie PII.
- notatki nie mają określonej struktury, może tam być cokolwiek.
- notatek nie można było edytować i usuwać.

Cel

Przychodzi managerka i mówi:

**Znajdź wszystkie notatki które
zawierają PII i zredaguj je!**

Context is the king!



Co mogę zrobić?

Jakie mam opcje?

- Ręcznie je sprawdzić?
- Regexy?
- SaaSy?
- Jakież libki?
- Samodzielnie ogarnąć AI/ML?
- Zmienić pracę?





Amazon Comprehend to the rescue!



**Amazon
Comprehend**



Funkcjonalności



Social media posts, emails,
web pages, documents,
phone transcripts and
medical records



Amazon Comprehend

Automatically extract key
phrases, entities, sentiment,
language, syntax, topics and
document classifications



Entities



Sentiment



Key Phrases



Syntax



Language



Topics



Document Classifications

Extracts data, topics, and document
classifications with confidence scores

Entities

Key phrases

Language

PII

Sentiment

Targeted sentiment

Syntax

Personally identifiable information (PII) analysis mode

☒ Offsets

Identify the location of PII in your text documents.

☐ Labels

Label text documents with PII.

Analyzed text

Hello Zhang Wei, I am John. Your AnyCompany Financial Services, LLC credit card account 1111-0000-1111-0008 has a minimum payment of \$24.53 that is due by July 31st. Based on your autopay settings, we will withdraw your payment on the due date from your bank account number XXXXXX1111 with the routing number XXXXX0000.
Customer feedback for Sunshine Spa, 123 Main St, Anywhere. Send comments to Alice at sunspa@mail.com.
I enjoyed visiting the spa. It was very comfortable but it was also very expensive. The amenities were ok but the service made the spa a great experience.

▼ Results

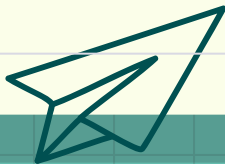

<div><div><div></div><div>Search</div></div><div><div><</div><div>1</div><div>></div><div></div></div></div>		
Entity	Type	Confidence
Zhang Wei	Name	0.99+



Typy danych



Entity Type	Entity Type	Entity Type
ADDRESS	DRIVER_ID	PHONE
AGE	EMAIL	PIN
AWS_ACCESS_KEY	INTERNATIONAL_BANK_ACCOUNT_NUMBER	SWIFT_CODE
AWS_SECRET_KEY	IP_ADDRESS	URL
CREDIT_DEBIT_CVV	LICENSE_PLATE	USERNAME
CREDIT_DEBIT_EXPIRY	MAC_ADDRESS	VIN
	PASSPORT_NUMBER	



Amazon Comprehend

Rodzaje detekcji

synchroniczna

- odpowiedź w czasie rzeczywistym
- limit znaków 5,000 na jeden request
- redagowanie tekstu we własnym zakresie

asynchroniczna

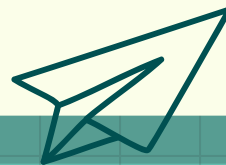
- background job
- Input i Output w S3
- 1MB tekstu na jeden request
- Możliwość automatycznego redagowania tekstu




Detekcja synchroniczna

ContainsPiiEntities

DetectsPiiEntities





```
import { Comprehend } from "aws-sdk";
...
const text = "My name is Daniel, my email
is example@example.com my pin number is 1234";

const params: PiiEntitiesRequest = {
  LanguageCode: "en",
  Text: text,
};
...
const detectPiiResponse = await comprehend
  .detectPiiEntities(params)
  .promise();

const containsPiiResponse = await comprehend
  .containsPiiEntities(params)
  .promise();
```

```
{
  "fullResponseContains": {
    "Labels": [
      {
        "Name": "PIN",
        "Score": 1
      },
      {
        "Name": "EMAIL",
        "Score": 1
      },
      {
        "Name": "NAME",
        "Score": 1
      }
    ]
  }
}
```

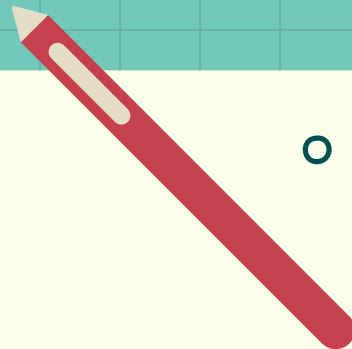
Różnica?

Offsety

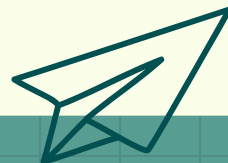
```
{
  "DetectPiiResponse": {
    "Entities": [
      {
        "Score": 0.9999493360519409,
        "Type": "NAME",
        "BeginOffset": 11,
        "EndOffset": 17
      },
      {
        "Score": 0.9999399185180664,
        "Type": "EMAIL",
        "BeginOffset": 31,
        "EndOffset": 50
      },
      {
        "Score": 0.9998846054077148,
        "Type": "PIN",
        "BeginOffset": 68,
        "EndOffset": 72
      }
    ]
  }
}
```



○ Redagowanie



**“In-house” odejmując od siebie
offsety, prosta operacja na stringu.**



Własne typy PII

Czyli co, jeśli Comprehend nie wspiera naszych typów PII

Entity Lists

- działa jak słownik, gdzie przekazujemy wszystkie możliwe wyrażenia.
- prosty i szybki, przez to nie do końca dokładny.
- mogą być bardzo precyzyjne, jeśli frazy, których szukamy, są dość unikalne i nie zmieniają się często.

Annotations

- bardziej elastyczny.
- uczy wzorców.
- wymaga znacznie więcej pracy.
- wymaga wielu zaawansowanych przykładów.

Pricing



Zgadnijcie jaka cena za
sprawdzenie 500 tysięcy notatek?

**Detects PII
Entities
250\$**



○ Jak liczyć koszty ○

○ 1 unit = 100 znaków

Price Per Unit				
Feature	Up to 10M units	From 10M - 50M units	From 50M - 100M units	Over 100M units
Detect PII	\$0.0001	\$0.00005	\$0.000025	\$0.000005
Contains PII	\$0.000002	\$0.000001	\$0.0000005	\$0.0000001



Przykład

10,000 notatek, każda zawiera 450 znaków.

4,5 miliona znaków = 45000 units (1 unit = 100 znaków)

Contains PII Entity:

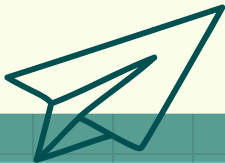
Cena za jeden unit = 0.000002\$

45000 units * (0.000002) = **0.09\$**

Detects PII Entity:

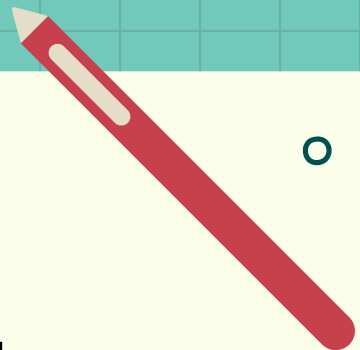
Cena za jeden unit = 0.0001\$

45000 units * (0.0001) = **4.5\$**



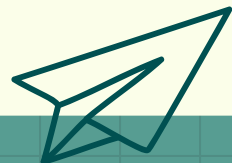


Demo App



Link: bitly.ws/FWGJ

Repo:
<https://github.com/Pigius/pii-service>



Demo App

Submit

My name is Dante

Creation date:

Detected PII entities:

- Dante is a NAME PII data type with a score of 0.9999828338623047

Redacted content:

My name is *****

Message length: 16

My pin number to credit card is as follow: 1234

Creation date:

Detected PII entities:

- 1234 is a PIN PII data type with a score of 0.983748197555542

Redacted content:

My pin number to credit card is as follow: *****

Message length: 47

Evaluation survey



Dziękuję!



Keep in touch:

Linkedin:
Daniel Aniszkiewicz