Copyright Takedowns Automation

Student: Mohammad Hossein Zarifi

Supervisor: Professor James Andrew Smith

Course: EECS 4088 (Fall/Winter 2021-2022)

Lassonde School of Engineering, York University, Toronto, Ontario, Canada, M3J 1P3

MOTIVATION

- There has been an increase in the number of course content being uploaded to commercial websites like Chegg and Course Hero.
- Copyright owners can submit a request to commercial websites, but the process is time-consuming and cumbersome.
- The purpose of this project was to automate the copyright takedown process in commercial websites.

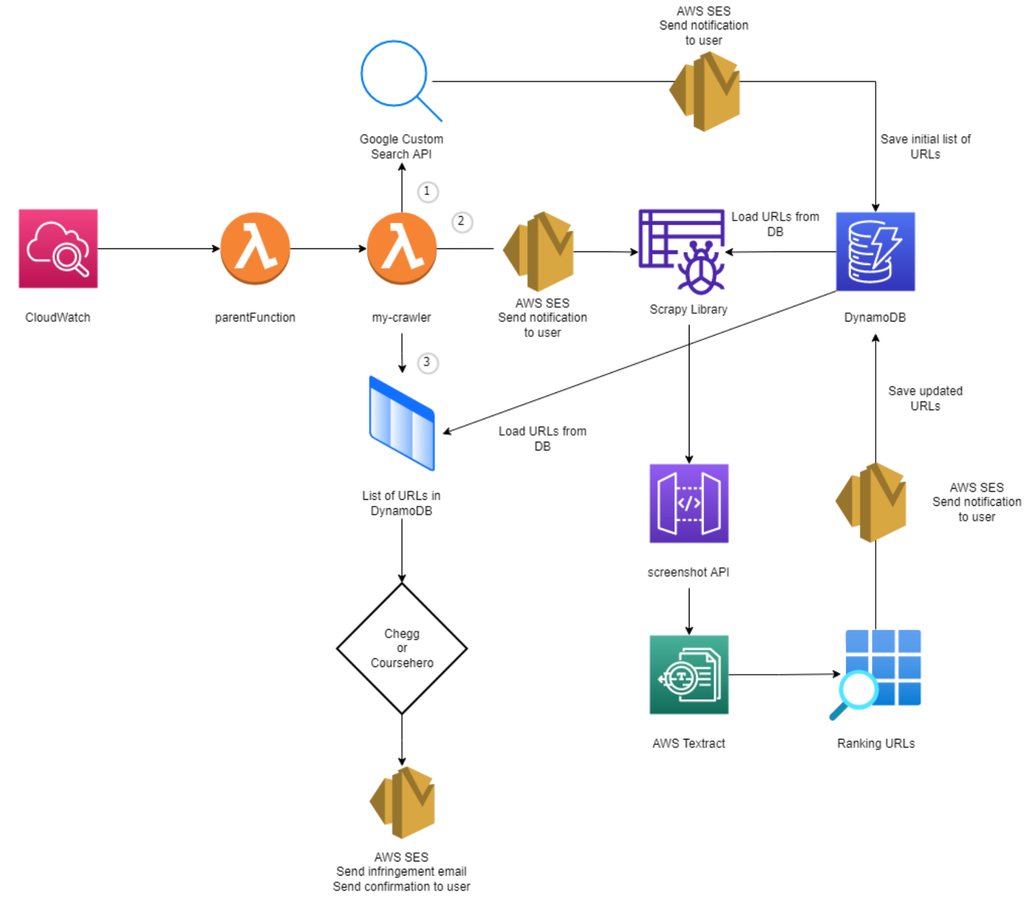
Solution

- A web application that consists of front-end and back-end modules that automates the process of crawling and takedown submission.
- The back-end is responsible for crawling the commercial websites and finding the URLs that might contain course materials that belong to the copyright owner.
- The front-end module enables users to see the result of the crawling and send commands to the back-end crawler. The commands that users can send include recrawling the commercial websites and submission of takedown requests.

Feature Highlights

- Use of serverless services such as AWS Lambda, AWS API Gateway, AWS S3, and AWS DynamoDB which reduces the operation and maintenance time.
- The combination of custom screenshot API and AWS Textract make it possible to analyze course materials in PDF format and determine their ownership.
- Copyright owners have the option to add keywords (via web application) to be searched by crawlers in the back-end. Users can take advantage of this feature and add a readable watermark to their course content which can make them easily identifiable by the back-end crawlers.
- Use of bot-evasion techniques in the crawler to avoid bot-detectors as much as possible.

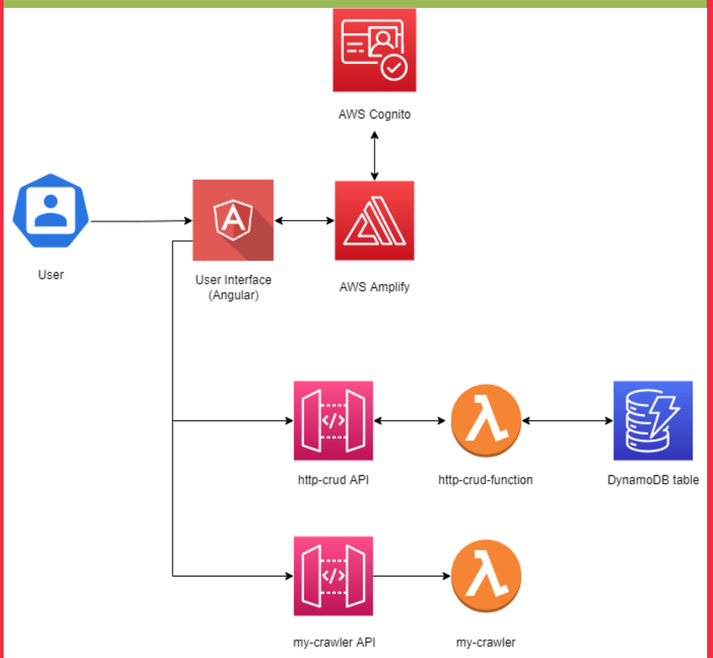
Back-End Architecture



How The Back-End Works

1. AWS CloudWatch is scheduled to call the Lambda function 'parentFunction' regularly and the 'parentFunction' asynchronously calls the second Lambda function 'my-crawler' to initiate the crawling of commercial websites. Google Custom Search API crawls the web pages of commercial websites and stores the target URLs in the AWS DynamoDB table.
2. The Lambda function 'my-crawler' has the logic to recrawl the commercial websites. In the recrawling stage the Python library Scrapy is used to scrap the commercial web pages. The custom screenshot API is used to take a screenshot from each URL that was found by the Scrapy library and the AWS Textract service is used to extract the keywords from the screenshot image. Any URL that is found to host copyright material (based on the extracted keywords) will be stored in the DynamoDB table.
3. The URLs stored in the DynamoDB table are used to submit the takedown request.

Front-End Architecture



How The Front-End Works

1. Front-End was written in Angular and it has been deployed to the public cloud via AWS Amplify.
2. AWS Amplify uses AWS Cognito to provide user registration and authentication.
3. A HTTP CRUD API (http-crud API) was created via AWS API Gateway. This API is used by the front-end to create/update/delete records in the DynamoDB table.
4. A REST API (my-crawler API) was created via AWS API Gateway. This API is used by the front-end to send different types of requests to the back-end crawler. The types of requests are as follows:
   - Recrawl
   - Submit takedown request

E-Mail | LinkedIn | Digital Poster