Project Report: TattScan

Joey Tatú  
15015556  
Cloud Platform Programming  
Postgraduate Diploma in Science in Cloud ComputingNational College of Ireland  
joey.tatu@student.ncirl.ie  
16th December 2021

*Abstract*—– a 150-300-word executive summary of the project and the main results

# Introduction

## Background

The inspiration for this project originated from the creation of a vegan health and fitness app. This project's concept was to take a snapshot or image of the eyes and provide a result depending on the sclera's whiteness or yellowness. Against support of the original concept, this application was expanded into an Android app that scans for diabetic retinopathy in the eye. [1] This project has a similar concept where a user can upload a photo of a tattoo and have the application identify it.

## Aims

Google Lens is a comparable image recognition system that can recognise a tattoo but not the pieces that make up a tattoo. Barcodes, QR codes, labels, text, famous buildings, and regions are all recognized by Google Lens. [2] [3]

TattScan works in a similar way to the diabetic retinopathy app. TattScan analyses and identifies a tattoo's primary elements, then develops a hashtag that it uses to search Twitter for comparable tattoos, which it then displays its results to the user. The user can decide if they wish to save the result.

This project has aimed to build a bridge between Google Lens and identifying elements of tattoos via image recognition. Currently, photo identification technology can identify tattoos, but not elements of a tattoo. This project aims to rectify this.

## Technology

TattScan is a web application that has been created using the Amazon Web Services (AWS) Cloud 9 integrated development environment (IDE), primarily using Python code. Possible future iterations will be developed for Android devices.

It was expected that Django would be used as a Python web framework. Comparably, a web site can be hard-coded, but tools are available to simplify this. This is like Flask which has been used instead. Flask removes the bulk, hard-coding aspect of a web framework and simplifies creating and deploying a web application. [4]

AWS Simple Storage Service (S3) has been used to store data files (objects) in containers (buckets). [5]

AWS DynamoDB has been used in thus project along with S3. AWS DynamoDB is a NoSQL database that remains consistent even when the database becomes larger or smaller. DynamoDB can also handle vast amounts of semi-structured data while adhering to rigorous latency constraints. [6]

These are the main aspects of this project in relation to storage. Further information on cloud services used in this project are in the Cloud-based Services section.

# Project Requirements

## Functional Requirements

The functional needs are ranked in this section. These describe the goals of the project.

### Requirement 1: Identify tattoo

This use case describes how a user uploads an image of a tattoo.

#### Description & Priority

This is the focus of this project. This use case explains how a user uploads an image of a tattoo and how the system identifies it. An option then is to allow the user to search the image on Twitter by using a hashtag.

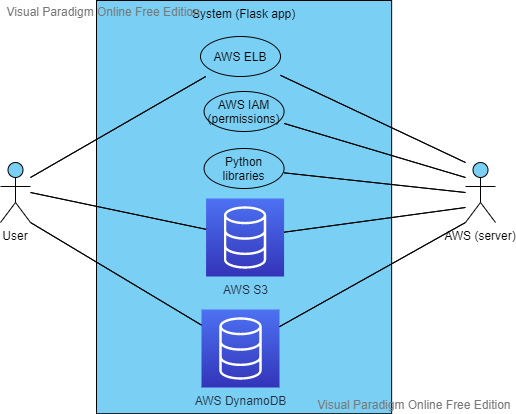
#### Use Case

Unique ID: identify-tattoo

**Scope**

The scope of this use case is for a user to upload an image and for the system to identify it as a tattoo and its main elements.

**Use Case Diagram**



(Larger images are at end of report for easy reading)

**Flow Description**

**Precondition**

* AWS and its services are active and running correctly.
* The Flask webapp is coded correctly and connected to the relevant services (e.g., AWS S3)
* Twitter and its API are active and running.

**Activation**

This use case begins when a user opens the Flask webapp.

**Main flow**

1. The user opens the Flask webapp.
2. The page loads with options to sign up, login and to upload a file. <See A1, A2>
3. The user selects upload a file

(External: User selects image from their device)

1. Image is stored temporarily in AWS S3
2. The relevant Python library identifies the image
3. The output if what it has identifies is outputted to the user.
4. The User clicks on one of the outputs. <See A3>
5. The System prefixes the output with a hash symbol and adds “tattoo” to the output to create a hashtag. (e.g. #starstattoo)
6. Using the Twitter API via a Python library, the System searches Twitter for images matching the generated hashtag.
7. The Systems then displays the results to the User.
8. The User clicks on an image   
   (External: User is taken to Twitter website)

**Alternative flow**

A1 <User logs in>:

1. The User clicks log in
2. The User enters their username and password
3. The System queries the DynamoDB to check credentials <See E1>
4. The User successfully logged in

<returns to Step 3 in Main flow>

A2 <User signs up>:

1. The User clicks sign up
2. The User enters their username and their password and their password again to confirm
3. The System create a new entry in the DynamoDB and encrypts User’s password.
4. The System logs in the User
5. The User successfully logged in

<returns to Step 3 in Main flow>

A3 <User doesn’t wish to search Twitter>:

1. The User decides not to click on an output.

<Use Case ends>

**Exceptional flow**

E1 < User’s username and/or password are incorrect>

1. The System outputs that the username and/or password are incorrect to the user.

<returns to Step 3 in Alternative flow: A1>

**Termination**

This use case is terminated when the User selects an output to go to the Twitter website.

**Post condition**

The system goes into a wait state.

## Non-functional requirements

### Data Requirements

Encryption is required for all data transmissions. It is not permissible to use raw data for communication from the Flask webapp to a service or Twitter API.

Passwords must contain at least one uppercase letter, one lowercase letter, one number, and one special character and must be at least eight characters long. There is a verification in place for this.

### User Requirements

Users should have a basic knowledge of browsing and searching online and uploading files to a website.

### Usability Requirements

The Flask webapp will be easy and straightforward to use. Accessibility features have been implemented such as being able to invert colours of the website to make it easier to read, as well as allowing the text to be made larger or smaller to aid those with visual impairments.

# Architectural Design Aspects

# 

(Larger images are at end of report for easy reading)

The Flask webapp may not appear to be very extensive at first glance but is detailed sub rosa.

AWS was chosen instead of other cloud platforms is because of AWS’s locations among other reasons. If one location fails, another one can be used. There are no restrictions on regional borders and one can use AWS from anywhere in the world. AWS also has a “disaster recovery system” and backup capabilities in different areas. Scalability is another important factor. AWS Auto-Scaling and AWS Elastic Load Balancing allow developers to scale their applications to their needs. [7]

AWS Identity and Access Management (IAM) is used for deployment of the Flask webapp to allow the webapp to be deployed and to monitor its health. AWS Elastic Beanstalk (ELB) is a solution for launching and scaling web apps and services. With using AWS, this was a logical solution to launch TattScan. [8] AWS Cloud9 is an IDE where the coding was completed. Creating the DynamoDB and AWS Simple Storage Service (S3) databases also were completed in Cloud9.

AWS Simple Queue Service (SQS) has been used in TattScan to ensure the server is not overloaded when a large group of people are using it.

Datadog has been used to monitor and analyse performance of TattScan. This is used to monitor whether more or less resources are needed.

{{Image processing Python library}}

{{Twitter Python library}}

# Cloud-based Services

Not shown in the diagram above is the AWS Elastic Compute Cloud (EC2) instances. The AWS Elastic Beanstalk, AWS Cloud9 and AWS Simple Storage Service (S3) all are an individual EC2 instance. It would be redundant to mention EC2 as a separate cloud service.

Two types of cloud databases have been used: AWS DynamoDB and AWS S3. DynamoDB is a NoSQL database with tables designed for storing structured data. [9] Therefore, it was chosen to store the account information. Because S3 can tolerate a high number of queries and unusual latency patterns, it was chosen to store the uploaded images. [9]

Technical experts and developers utilise AWS SQS to transmit, store, and retrieve many messages of varying sizes asynchronously. [10]

AWS SNS is an automatic service for messaging for communication between applications and people. [11]

Datadog is an online cloud service that is for monitoring and an analytics platform for determining performance metrics, as well as event tracking for infrastructure and cloud services. Servers, databases, and tools may all be monitored with the software. [12]

# Library Description

After the text edit has been completed, the paper is ready for the template. {{Custom Python library}}

# Implementation

After the text edit has been completed, the paper is ready for the template. Duplicate the template file by using the Save As command, and use the naming convention prescribed by your conference for the name of your paper. In this newly created file, highlight all of the contents and import your prepared text file. You are now ready to style your paper; use the scroll down window on the left of the MS Word Formatting toolbar.

# Integration, Delivery and Deployment

After the text edit has been completed, the paper is ready for the template. Duplicate the template file by using the Save As command, and use the naming convention prescribed by your conference for the name of your paper. In this newly created file, highlight all the contents and import your prepared text file. You are now ready to style your paper; use the scroll down window on the left of the MS Word Formatting toolbar.

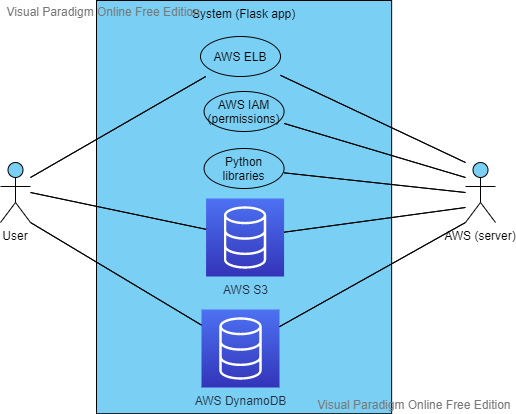
# Conclusions

Possibly talk about identifying issues with healing tattoos.

# References

|  |  |
| --- | --- |
| [1] | J. Tatú, “Automatic Retinopathy Detection Using Digital Image Processing via a Smart Device,” National College of Ireland, Dublin, 2021. |
| [2] | R. Nieva, “Forget rainbow vomit. Google Lens is AR you can actually use,” CNET, 18 May 2017. [Online]. Available: https://www.cnet.com/tech/mobile/google-lens-is-augmented-ar-you-can-actually-use-snapchat-facebook-instagram/. [Accessed 1 December 2021]. |
| [3] | A. Villas-Boas, “Google's new Lens product uses your phone's camera to do clever tricks, like connecting your phone to a WiFi network,” Business Insider, 17 May 2017. [Online]. Available: https://www.businessinsider.com/googles-lens-feature-can-connect-your-phone-to-wifi-using-your-camera-2017-5?op=1&r=US&IR=T. [Accessed 1 December 2021]. |
| [4] | S. Campbell, “Flask vs Django: What’s the Difference Between Flask & Django?,” Guru99, 1 November 2021. [Online]. Available: https://www.guru99.com/flask-vs-django.html. [Accessed 1 December 2021]. |
| [5] | Amazon Web Services, “What is Amazon S3,” Amazon Web Services, 1 December 2021. [Online]. Available: https://docs.aws.amazon.com/AmazonS3/latest/userguide/Welcome.html#CoreConcepts. [Accessed 1 December 2021]. |
| [6] | DynamoDB Guide, “What is DynamoDB?,” DynamoDB Guide, 15 December 2018. [Online]. Available: https://www.dynamodbguide.com/what-is-dynamo-db/. [Accessed 15 December 2021]. |
| [7] | S. Bhatt, “7 Reasons to Choose AWS as Your Cloud Platform,” DZone, 31 July 2021. [Online]. Available: https://dzone.com/articles/7-reasons-to-choose-aws-as-your-cloud-platform. [Accessed 13 December 2021]. |
| [8] | Amazon, “AWS Elastic Beanstalk,” Amazon, 13 December 2021. [Online]. Available: https://aws.amazon.com/elasticbeanstalk/. [Accessed 13 December 2021]. |
| [9] | N. Samuel, “Amazon S3 vs DynamoDB: 5 Critical Differences,” Hevo Data, 8 November 2021. [Online]. Available: https://hevodata.com/learn/amazon-s3-vs-dynamodb/#factors. [Accessed 13 December 2021]. |
| [10] | Amazon, “Amazon Simple Queue Service,” Amazon, 13 December 2021. [Online]. Available: https://aws.amazon.com/sqs. [Accessed 13 December 2021]. |
| [11] | Amazon, “Amazon Simple Notification Service,” Amazon, 13 December 2021. [Online]. Available: https://aws.amazon.com/sns/. [Accessed 13 December 2021]. |
| [12] | A. S. Gillis, “Datadog,” SearchITOperations, 13 December 2021. [Online]. Available: https://searchitoperations.techtarget.com/definition/Datadog. [Accessed 13 December 2021]. |

# Appendix



This is an image of the Use Case and how a user interacts with the System.

Diagram

Description automatically generated

System Architecture design.

A screenshot of a computer

Description automatically generated

Code showing the creation of the S3 Bucket and running the server in AWS Cloud9

Graphical user interface, text

Description automatically generated

Setting up and checking app routes (pages) are working in the webapp.

A screenshot of a computer

Description automatically generated with medium confidence

Setting up AWS DynamoDB and testing with a batch-write file that adds 3 entries.