2022

iTag: A Modern Image Storage on the Cloud

FIT5225 Assignment 2 Team 10 Report

STUDENT ID: 31303501 STUDENT NAME: MINGXUAN CHEN

STUDENT ID: 31524605 STUDENT NAME: YUQI XIE

STUDENT ID: 31093833 STUDENT NAME: YANAN WU

STUDENT ID: 30392020 STUDENT NAME: ZIXUAN LEI

Contents

Table of Figures 2
Table of Tables 2
1 Introduction 3
2 AWS Architecture diagram 3
2.1 How Does the Application Working 4
2.1.1 Hosting 4
2.1.2 Authentication 4
2.1.3 API 4
2.1.4 Data Storage 5
2.1.5 Cloud Watch 5
3 Description of team members 6
4 User Guide 6
4.1 Authentication and Authorisation 6
4.2 Image Upload 7
4.3 Queries 9
4.3.1 Search images by tags 9
4.3.2 Search images by images 9
4.3.3 Manual addition or removal of image tags 11
4.3.4 Delete an image 12
4.4 Application Security 13
5 Source Code Link 13
6 Bibliography 13

Table of Figures

Figure 1 AWS Architecture Diagram 3
Figure 2 Login Interface 7
Figure 3 Welcome Page 7
Figure 4 Image Upload Page 8
Figure 5 Image Detec Success 8
Figure 6 Message of Image Upload Success 8
Figure 7 Search by Tag Page 9
Figure 8 Results of Search 9
Figure 9 Search by Image: Image Upload Page 10
Figure 10 Image Detect Result 10
Figure 11 Results of the Search 11
Figure 12 Management Dashboard 11
Figure 13 Edit Interface 12
Figure 14 Updated Tag 12
Figure 15 Delete Image Confirmation Panel 12
Figure 16 Image Delete Success 12

Table of Tables

Table 1 API Gateway and Lambda Function association with the data storage 4
Table 2 Team Contribution Table 6

iTag: A Modern Image Storage on the Cloud

FIT5225 Assignment 2 Team 10 Report
www.iweblens.tk

1 Introduction

With rapid development in the information technology field, the needs for fast-developing are stronger and stronger. The serverless application can shift the stress of server maintenance and configuration from the developers to the service provider, while also providing the version control functionality to the developer (Juan A. Añel, Diego P. Montes, Javier Rodeiro Iglesias). This report is written to show the design architecture and usage of an image storage cloud application designed and delivered by Monash University FIT5225. The final presentation can be found on *www.iweblens.tk*.

2 AWS Architecture diagram

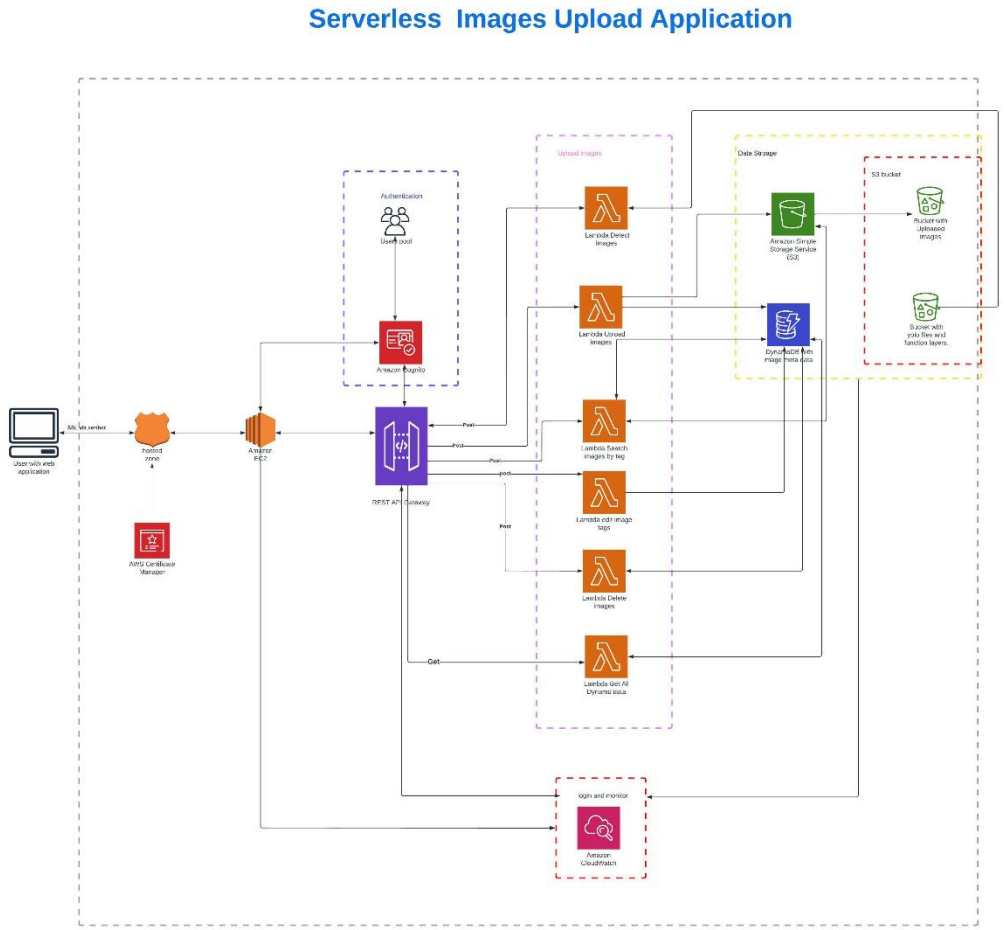

Figure 1 AWS Architecture Diagram

2.1 How Does the Application Working

2.1.1 Hosting

2.1.1.1 Nginx

The website is hosted on an EC2 server as a static website, among Nginx and Apache, we decided to use Nginx as our proxy server since it is easy to configure, high performance, and can work with Certbot to auto-generate SSL encryption for the website.

2.1.1.2 Route 53

As shown in Figure 1, Route 53 is the first layer of the whole structure. The domain of the website is registered under Freenom, however, Route 53 is used to generate the Name Server and save the DNS records. It allows the user to modify their domain upon requirements.

2.1.1.3 Amazon Certificate Manager (ACM)

Amazon Certificate Manager (ACM) is the Amazon official facility to manage the website associated certification. In this practice, although it's not shown in Figure 1, it is worked as a middleware component to use SSL to secure the connection between the Cognito endpoint and website.

2.1.1.4 EC2 Instance

The EC2 instance is chosen to host the front-end static website. The file was dropped onto the instance by using WinSCP. SSH is required to log in as part of security practice. Putty is used to configuring the Nginx service. The server instance allocated is AWS t2.micro free tier.

2.1.2 Authentication

2.1.2.1 Cognito

Cognition is the AWS authentication service that we have invoked to secure our service. One user pool is created to store the user information. Each action under the application will be requested to show the credential of the login.

2.1.3 API

2.1.3.1 API Gateway

Since the server is deployed on the serverless service, the endpoint of API is no longer the website domain, therefore, API Gateway is created to specify the endpoint of each resource and method. The Cognito authorizer is applied to check the access. All the Gateway resources are associated with lambda functions in section 2.1.3.2.

2.1.3.2 Lambda Function

As the major logical back core of the application, several lambda functions were created to communicate with the data source and answer client requests. Six Lambda functions are created in total, and their functionality can be shown in

Table 1 API Gateway and Lambda Function association with the data storage

API Gateway Method	Lambda Function Name	Brief Function
POST	iWebLens	Use S3 deployed layers (OpenCV and Yolo Config) to detect image content labels and then return to the frontend.

API Gateway Method	Lambda Function Name	Brief Function
POST	imageUpload	Receive the base64 format image string and associated label and UUID from the frontend as JSON and then append them into the Dynamodb (UUID as primary key, where label list and s3url associated) and S3 bucket (base64 string write to the image on S3)
POST	seachByTag	Receive tag as a list in JSON from front end and return associated pictures S3 URL and the base64 string which read from S3
POST	changeTag	Receive a new record with include UUID, new tag list and s3url and change it on Dynamodb records
POST	imageDelete	Receive the UUID and S3 URL and remove the associated record from Dynamodb and S3
GET	imageScan	Get all the image records from Dynamodb and the associated base64 string which read from S3

2.1.4 Data Storage

2.1.4.1 S3 Buckets

S3 Buckets is a simple file storage platform on AWS. In this practice, three S3 buckets are created to store files. The first two are created to save OpenCV dependencies and YOLO tiny configurations. These two buckets are bound with the iWebLens lambda service to serve the dependencies. And the last one is created to save the images as files.

2.1.4.2 Dynamodb

AWS served NoSQL database service. Three attributes are created: id: string, tag: list, and s3url: string. These three attributes are saved synchronized with S3 bucked contents.

2.1.5 Cloud Watch

AWS also provides a real-time monitoring service, the usage of each service and resource can be monitored in a graphic system. Logs are temporally also stored on the system which can provide debug information to the developers.

3 Description of team members

Table 2 Team Contribution Table

Student ID	Student Name	Contribution	Contribution Element
31303501	MINGXUAN CHEN	25%	Back-end, Lambda, change tag Axios call, part of Front-end issue fix, Front-end configuration, front-end merge and build, Team report.
31524605	YUQI XIE	25%	Front-end web pages with navigation and other click events, Team report.
31093833	YANAN WU	25%	Help with Back-end detection lambda function, get all dynamo image lambda function, Cognito, dynamo DB setup, ec2 and Nginx configuration, Team report.
30392020	ZIXUAN LEI	25%	Interact with the backend to send ajax requests. Team report.

4 User Guide

4.1 Authentication and Authorisation

The user first goes to the login screen and then uses the correct username and password to access the application home page. The application home page is the welcome interface, which briefly describes the functions of the Serverless Image Upload Application. Users can implement functions through the navigation bar on the left.

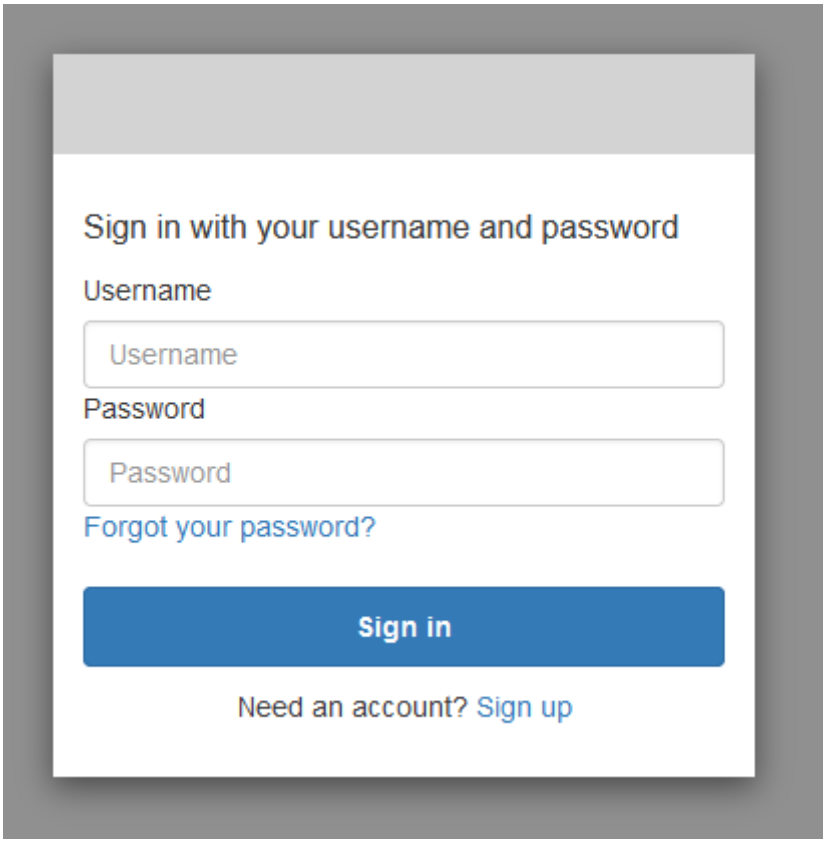

Figure 2 Login Interface

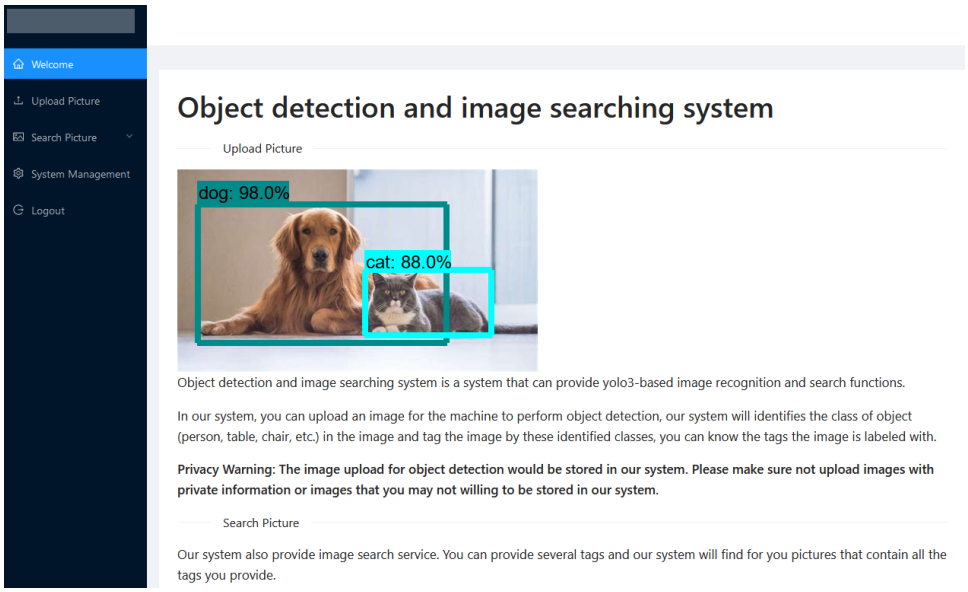

Figure 3 Welcome Page

4.2 Image Upload

Users can directly drag and drop the target image to upload to the application through the upload interface. After uploading the image, the app will confirm to the user whether to submit it.

e

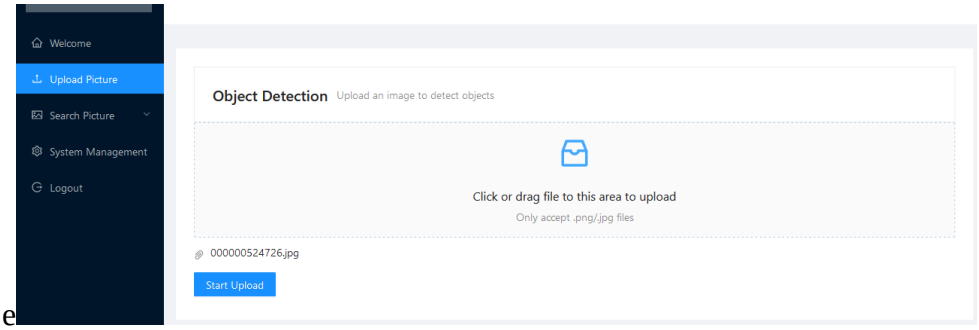

Figure 4 Image Upload Page

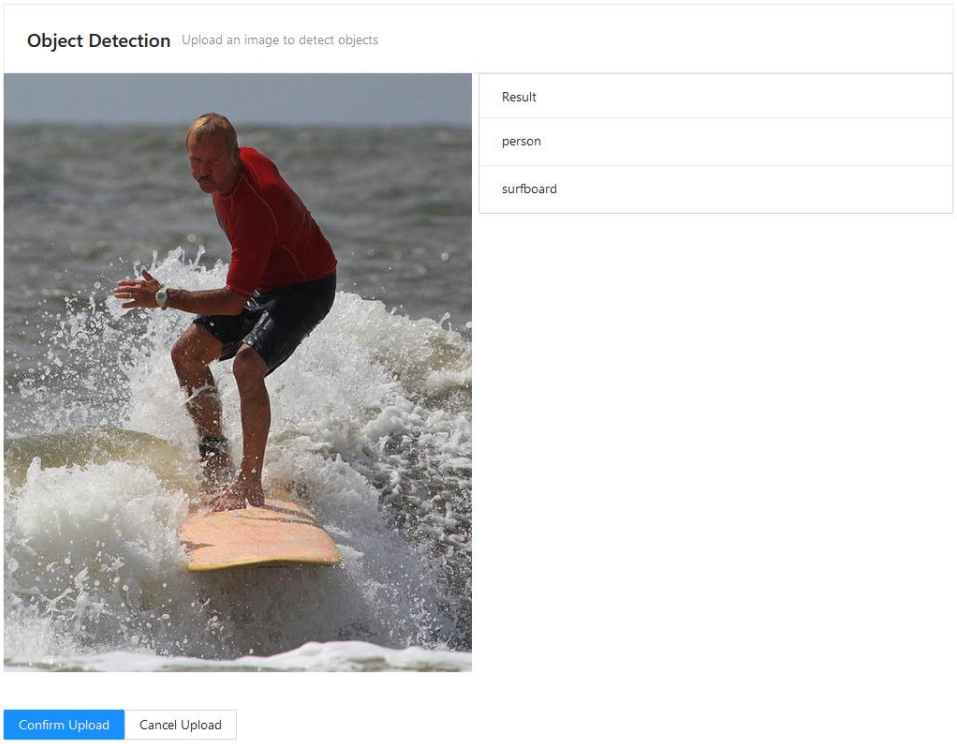

Figure 5 Image Detec Success

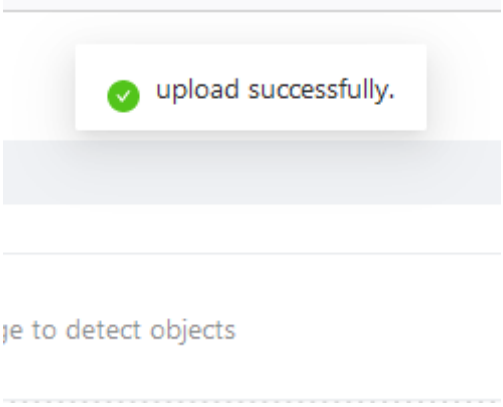

Figure 6 Message of Image Upload Success

4.3 Queries

4.3.1 Search images by tags

The application allows users to search for images in the storage space using tags, and the search results will be presented to the user in the form of a marquee.

Results:

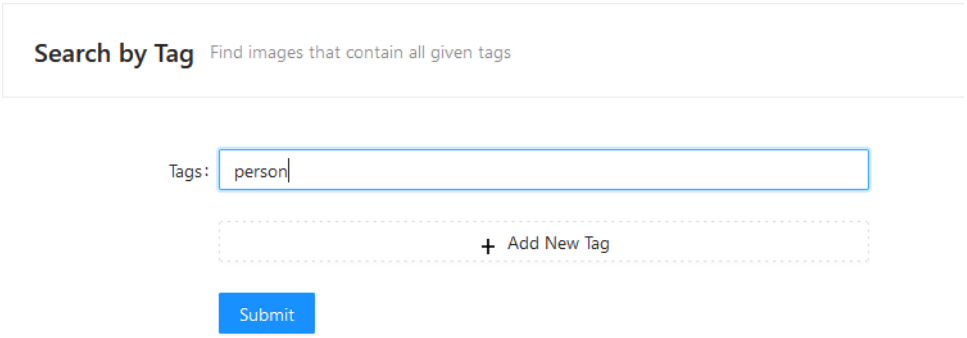

Figure 7 Search by Tag Page

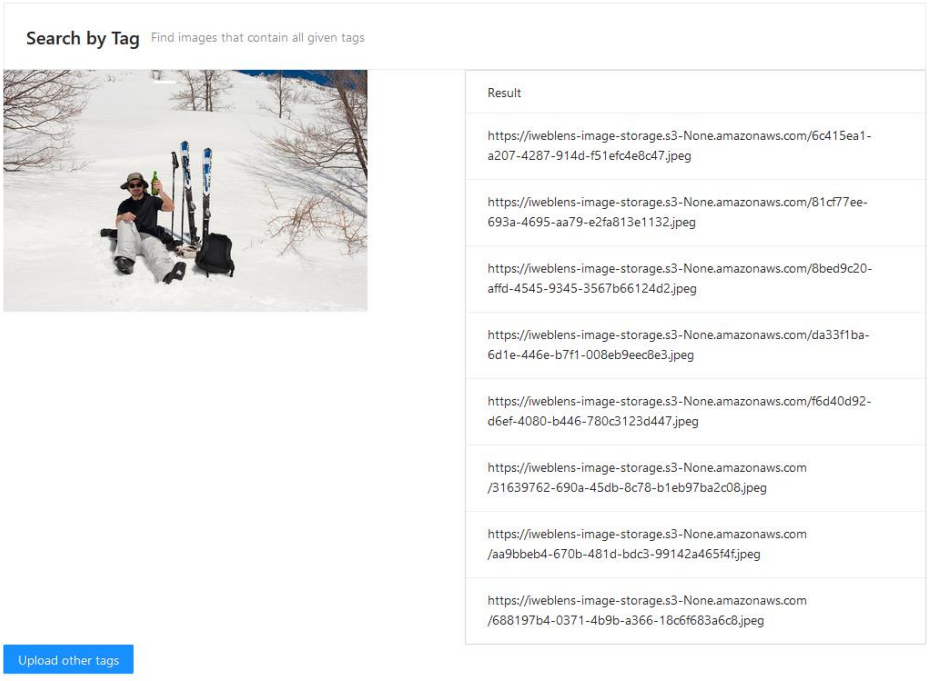

Figure 8 Results of Search

4.3.2 Search images by images

The application allows users to search for images in the storage space by image. Users can upload images to the search box, and the application will automatically read the tags of the images and search for images based on the storage space.

Results:

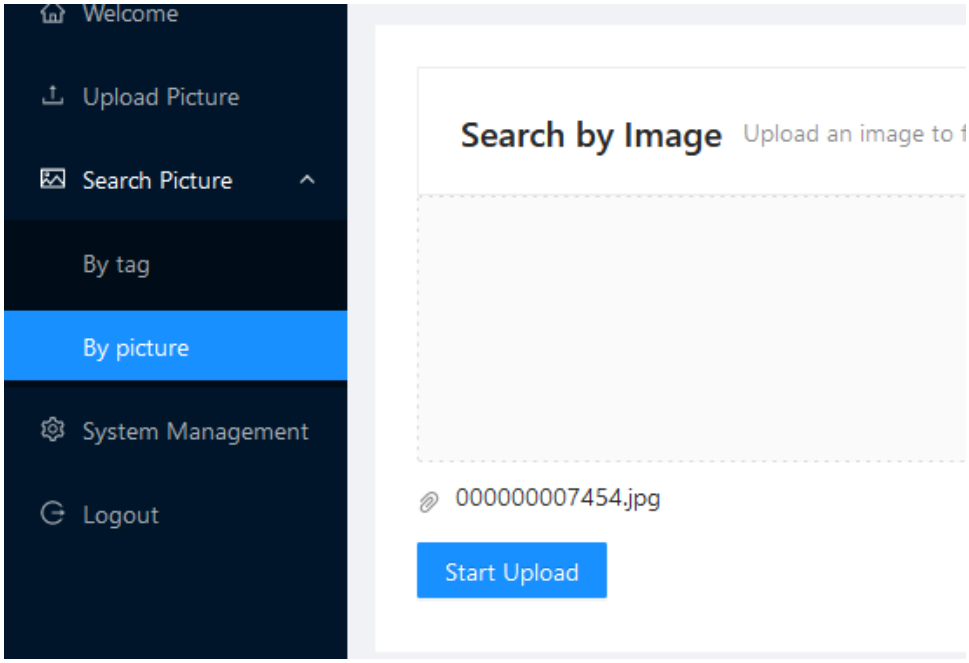

Figure 9 Search by Image: Image Upload Page

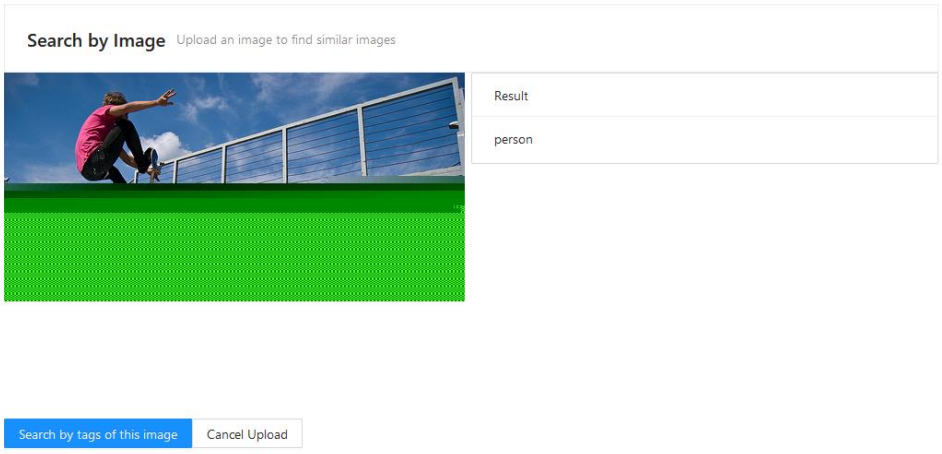

Figure 10 Image Detect Result

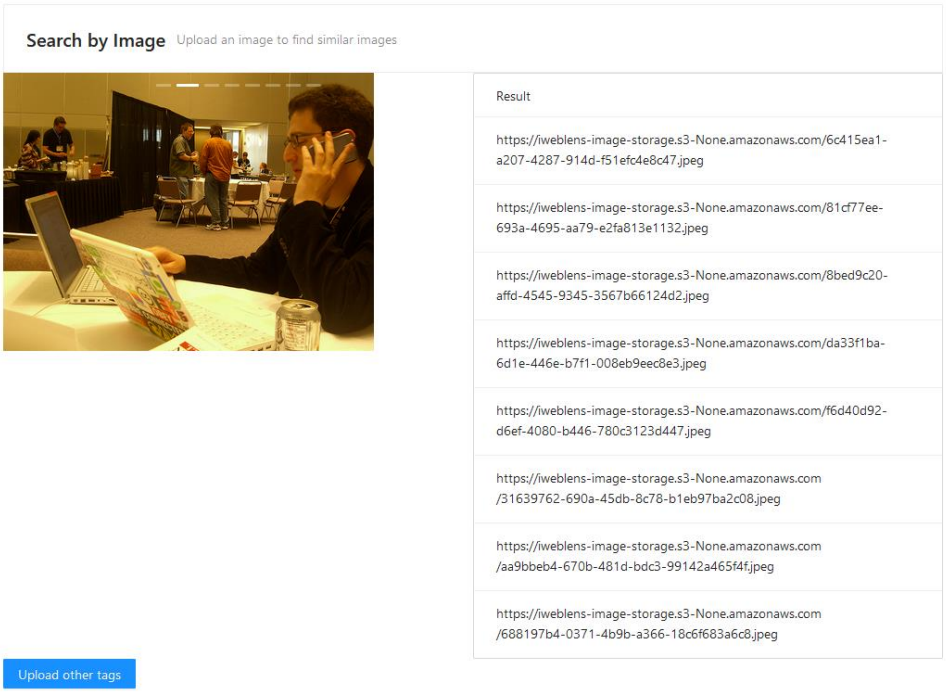

Figure 11 Results of the Search

4.3.3 Manual addition or removal of image tags

Users can add or delete tags for images.

Results:

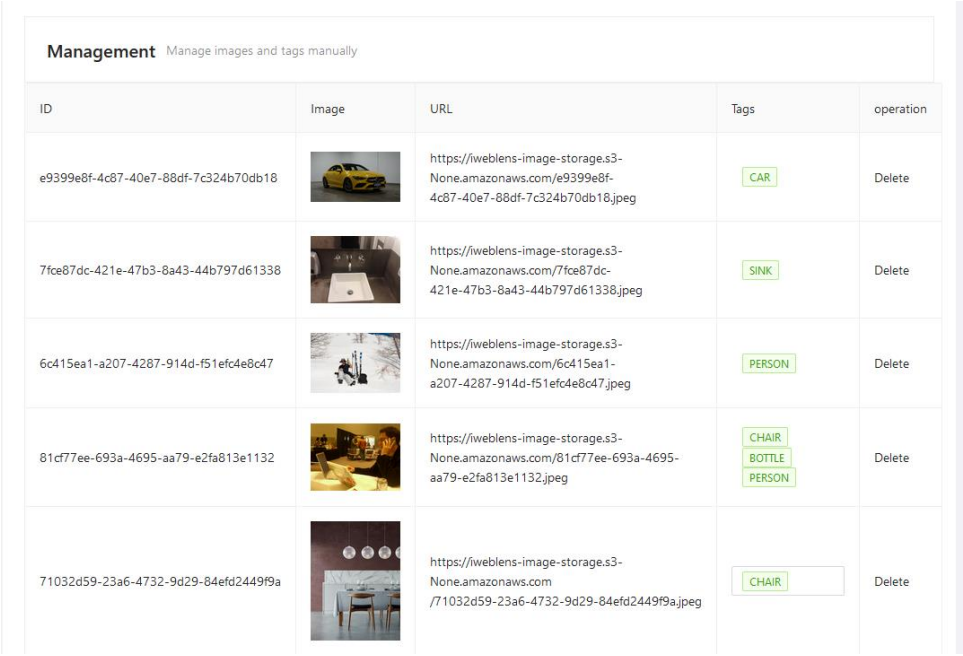

Figure 12 Management Dashboard

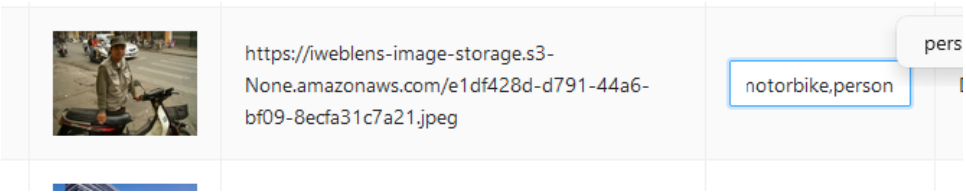

Figure 13 Edit Interface

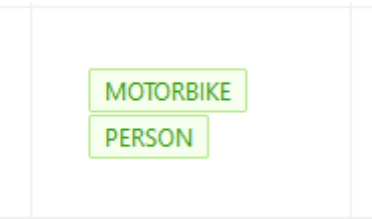

Figure 14 Updated Tag

4.3.4 Delete an image

Users can delete their uploaded images by clicking the delete button and then confirming.

Results:

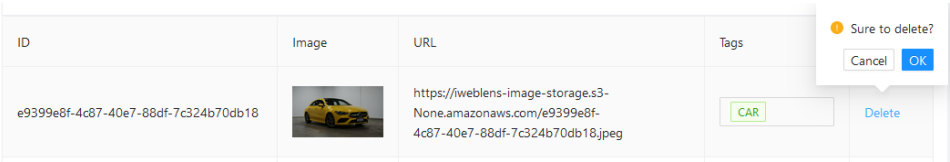

Figure 15 Delete Image Confirmation Panel

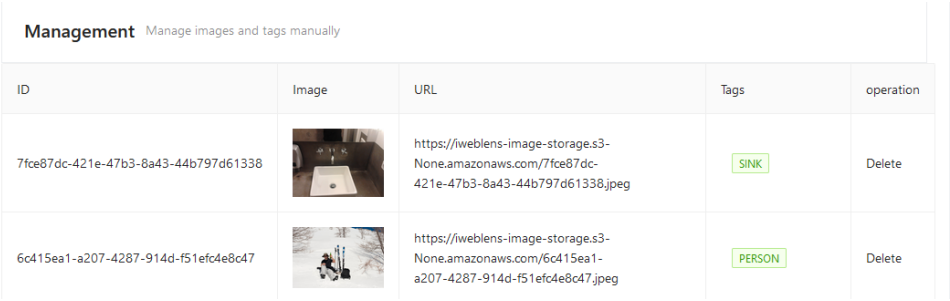

Figure 16 Image Delete Success

4.4 Application Security

Enhancing data security is about maintaining the privacy of individuals and preventing unwanted sources from affecting the data. The business value of data has never been greater today, so trustworthiness is increasingly important to consumers.

To protect customers' data, this application requires users to log in and authenticate before accessing storage. After successful login, users can upload images, retrieve images, view retrieval results, and cancel login operations.

5 Source Code Link

https://github.com/mche0126/fit5225-assignment2-front

6 Bibliography

Juan A. Añel, Diego P. Montes, Javier Rodeiro Iglesias. (n.d.). *Cloud and Serverless Computing for Scientists.* Springer Nature Switzerland AG 2020.