Name: Nicholas Erokhin
Title: Image tag search
Project Summary: The Image Tag Search project will allow a user to search for images meeting a certain criteria by their specified tags. Google cloud vision API and/or Amazon Rekognition will be used to generate the image tags. Additionally, the user will be able to search for text in images as well. All image tags will be indexed in order to perform the queries as fast as possible, and image names will be displayed in order with the strength of relevance to the query. This will be a command-line run program supplied with arguments.

Project Requirements:

1.x – User interaction with system

ID	Requirement Text
1.1	The system shall accept user uploads
1.2	The system shall allow users to download one or more images
1.3	The system shall provide the ability for a user to add and remove tags from an image
1.4	The system shall provide the ability to the user to list images in a system
1.5	The system shall provide the ability for the user to search for image tags and image text
1.6	The system shall provide the ability to the user to view the image without downloading to disk
1.7	The system shall allow the user to provide tags when specifying the image to upload
1.8	The system shall allow the user to delete images
1.9	The system shall allow the ability for the user to index every image
1.10	The system shall allow the ability for the user to batch-upload image files with a CSV file

2.x – Program behavior behind-the-scenes

ID	Requirement Text

2.1	The system shall index every new upload automatically unless specified otherwise
2.2	The system shall provide an index on image text and image labels
2.3	The system shall persist the image index on disk
2.4	The system shall load the image index on program launch
2.5	The system shall rank search results based on query relevance
2.6	The system shall return search results with the image name and image tags side-by-side
2.7	The system shall return search results in a timely manner

User Interface Mockups:

```
Parsing csv...

Successfully uploaded image dinner_party.jpg to s3
Successfully uploaded image patron.jpg to s3
Successfully uploaded image patagonia.png to s3
Uploading image imgas123.png failed, image does not exist


nicks-mbp:Desktop nickerokhin$ imgSearch --upload-csv ./ims.csv
```

```
ALL IMAGES:

Name        Location
----        --------

fish.png    s3
pond.jpg    local
img_322.jpg local/s3
1231ras.jpg local/s3

33 more...


nicks-mbp:Desktop nickerokhin$ imgSearch --list-images
```

```
QUERY: fish, water THRESH: 0.3

RESULTS:

Returned 5 results

NAME:          LOCATION:       RELEVANCE:
-----          ---------       -----------
fish.png       s3              100
pond.jpg       local           93
dinner.jpg     local           63
.....


nicks-mbp:Desktop nickerokhin$ imgSearch --search fish,water --thresh 0.3
```

Class Diagram

Document

+ documentPath: string
- locallyHosted: boolean
- cloudHosted: boolean
+ name: string

+ getDocumentPath(): string
+ setDocumentPath(): string
- isLocallyHosted(): boolean
- isCloudHosted(): boolean
+ getName(): string
+ setName(string: name): string

DocumentController

+ documents: Images[]

+ addDocument(docPath: string)
+ removeDocument(docName: string)
+ uploadDocument(docName: string)
+ getTagsFromCloud(docPath: string)
+ uploadCSV(csvPath: string)
+ downloadImage(docPath: string)
+ viewImage(name: string)

Command

+ commandType: string

+ execute(command: string):

Image

+ tags: string[]

+ getTags(): string[]
+ setTags(string[])
+ addTags(string[])

UserController

+ arguments: string[]

+ getArguments(): string[]
+ setArguments(string[])
+ parseArguments(string[]):

DocumentVector

s: string[]
agePath: string
ctor: int[]
ctorSize: int
evanceThreshold: int

Tags(): string[]
Tags(tags string[])
ImagePath(imagePath: string)
ImagePath()
Vector()
ateVectorFromTags(tags: string[])
Vector()

SearchIndex

+ documentMatrix: int[][]
+ columnSize: int
+ documentCount: int
+ documentMatrixMap: map<matrixIndex int: documentName string>
+ documentMap: map<documentName string: Image document>
+ tagVectorIndexMap: map<tag: string: vectorIndex int>
+ threshold: float

+ addDocument(vector: int[])
+ addDocumentVector(vector: DocumentVector)
+ addTagToIndex(tag: string)
+ removeTagFromIndex(tag: string)
+ getColumnSize(): int
+ setColumnSize(int size)
+ getDocumentCount(): int
+ setDocumentCount(int count)
+ search(queryVector): matrixIndices int[]
- saveMatrix()
- loadMatrix()

SearchResults

+ numResults: int
+ queryTags: string[]
+ queryResults: DocumentVector[]
+ threshold: float

+ getNumResults(): int
+ setNumResults(numResults: int)
+ getQueryResults(): DocumentVector[]
+ setQueryResults(DocumentVector[]):
+ setThreshold(threshold: float)
+ getThreshold(): float
+ filterResultsByThreshold(threshold: float): DocumentVector[]
+ displayResults()