Functional Specification Contents

0. Table of contents

1. Introduction

1.1 Overview

In the modern era it is standard that everyone has a camera attached to their phone that they carry with them everywhere they go. And with the rise of social media we can see more and more people taking photos of things and posting them online. With this vast source of information being generated every minute of every day it is hard to regulate all the information that is being generated. A human cannot do this and so an automated solution is necessary to identify images of copyright and find out if they are in breach of rules and regulations.

One problem is even though artwork is in public domain, the photograph of that painting is not in the public domain. The image belongs to the art gallery or in some cases the photographer. This can be heavily debated and has been for a number of years but I believe that even though I have narrowed down the domain, the work that will be done on this project will be useful across multiple domains.

There are many instances across the internet where images of artworks are being used without the galleries permission, taking from the revenue of the gallery. I am going to use this sample problem to attempt to find paintings in images that have been sourced from the internet.

These images may take many forms, some examples being, people trying to hide the fact that they are using the images and changing the hue of the image, cropping the image or making small transformations on the image to ensure that an automated way of detecting these images would not succeed, or be too computationally difficult to find across the internet at scale.

This project will explore creating a system that can be left to scour the internet in search of copyright breaches and notify parties that there is a copyright breach occuring. This solution should also have the capability to be fast enough for an end user to want to use the system and have the results returned to them in a reasonable time frame.

1.2 Glossary

OpenCV: OpenCV is a library of programming functions mainly aimed at real-time computer vision

Lambda Function: AWS Lambda is an event-driven, serverless computing platform provided by Amazon as a part of the Amazon Web Services. It is a computing service that runs code in response to events and automatically manages the computing resources required by that code.

S3 (S3 Bucket): Amazon Simple Storage Service (Amazon S3) is an object storage service

SIFT: Scale-Invariant Feature Transformation is a feature detection algorithm in computer vision to detect and describe local features in images.

2. General Description

2.1 Product / System Functions

This product will be a unique system that has two distinct parts. The first part of this system will be concerned with creating a database of images that are sourced from the wider internet. It will start with well known sites that are likely to be hosting artworks because these are the sites that are going to have most users viewing the enfriging images and the most revenue is stood to be lost for the gallery. A web scraper will be collecting these images and storing the data generated by the algorithm in a data store completely unsupervised. The web scraper will have the functionality to only store images that it detects has paintings in them to ensure that the data in the data store is relevant and worth while searching.

This data may look like key point lists from the images and metadata about the images. It will be compiled into a data object and stored in a highly available could data store service called Amazon S3. With precomputation and this extremely fast and professional datastore this will allow match finding very fast.

The second part of this system is the part that the user interacts with, labeled the front end of the system. This part of the system is concerned with taking an image from the user and handing it off to be processed. This will also return a data object that can be matched in the data store. This is a fast way to attempt to match results in the data store without having to compare the image to a every other image in the database. We hope that this will allow the user to get an almost instant result.

2.2 User Characteristics and Objectives

The completed product would mean it is continually scraping the web looking for images that have paintings in them. The system will be smart enough to only store images that contain paintings otherwise we will have tonnes of data that is not relevant at all that will take up far more space and time then we need. Once the web scraper module finds an image, it will process them into textual data to compare them at a later time. Metadata will be stored in a data store also. This can be extra data about the image such as gps location or semantic data like where on the web it was found. This data store would have precomputed data because that will speed up the search considerably.

This data store will be used to facilitate a search interface where by the user of the system is able to query this data store with another image of a painting. The search system will work to lookup this artwork and see if the database has this artwork contained in the images.

2.3 Operational Scenarios

The main operational scenario in this project will involve a user navigating to a standard and very simple web application. This application will be deliberately very plain to emphasize the simplicity of the system.

Once on the homepage of the web application the user will upload a single image to the system. This system will the give the user feedback to as to whether the image has been successfully uploaded or in fact there was an error on upload. The user will also be given feedback that there is background processes happening as there will be an expected delay between when the image is uploaded and the data being returned.

The information will be returned in the form of textual data. The user will be presented with a result that will indicate that there was no match found for this image in the database or it will indicate that there is infact a match and how similar that match is. It will include a web link to where the image was originally found should the user want to follow the image came to its source.

Should the user require another search of the system, the user will click a button to return them to the home screen or they may scroll to the top if the page they are currently on and perform another search of the system. There will not be any functionality to make a search of multiple images in the scope of this project however that is a functionality that could be implemented at a later date.

From a user perspective this scenario is the only scenario that will occur as will be reflected in the use case. The system is being designed with one purpose in mind and will hopefully achieve that process to a high standard.

User Jo navigates to web application:

- Current System State: Jo opens web browser
- Informal Scenario: Jo goes to correct URL to find web application
- Next Scenario: Jo wants to search the database for an image

User Jo wants to search the database for an image:

- Current System State: Jo on home page of web application
- Informal Scenario: Jo clicks search box and uploads an image from their computer to search the database for
- Next Scenario: Jo expects to receive results from search

User Jo expects to receive results from search:

- Current System State: Jo has uploaded an image to the search system
- Informal Scenario: Jo waits for the search indication to leave the screen and the results to appear
- Next Scenario: Jo receives results

User Jo receives results:

- Current System State: Jo is waiting on results
- Informal Scenario: Jo sees results appear on the screen in a text format
- Next Scenario: Jo wants to make a new search

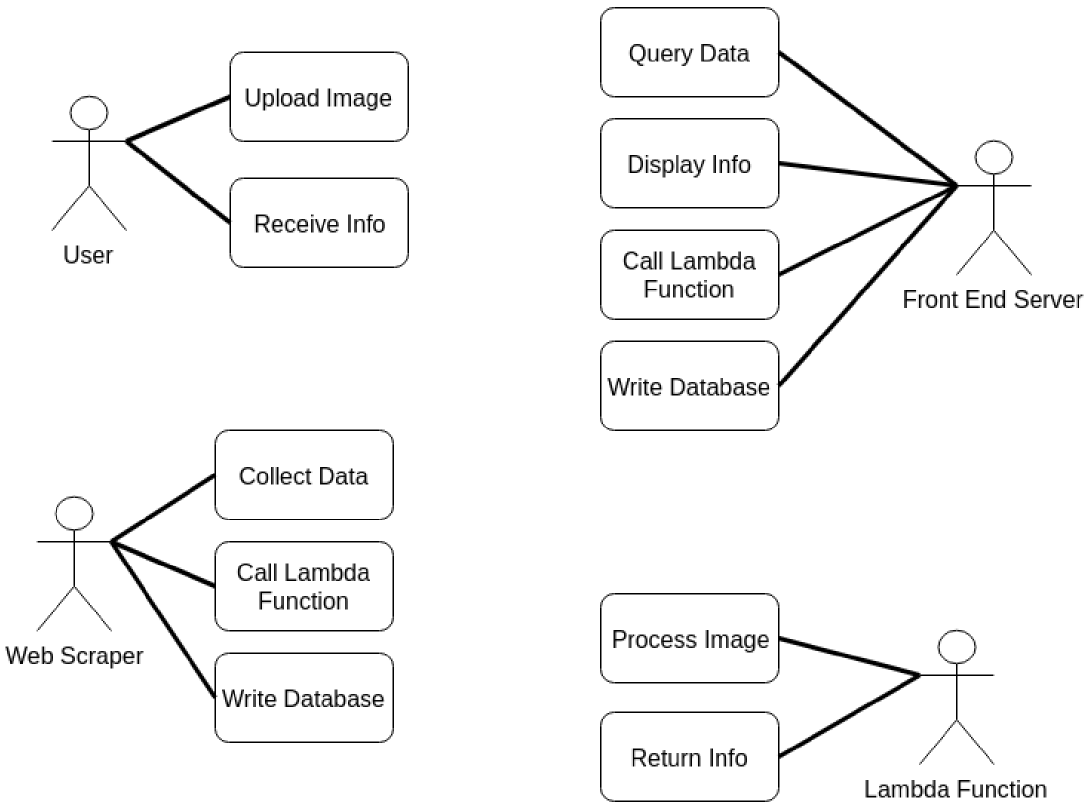
User Jo wants to make a new search:

- Current System State: Jo has received results
- Informal Scenario: Jo scrolls to the top of the page and uploads a new image in the search area
- Next Scenario: Jo waits on results

User Jo encounters and error:

- Current System State: Jo is waiting on image search results
- Informal Scenario: Jo sees an error message informing them that the image failed to upload
- Next Scenario: Jo wants to make a new search

Use Case 1	User Uploads Image
Goal in Context	User wants to recieve data about image
Scope & Level	User interface
Preconditions	Web server is operational and serves the user interface Database is online and operational Functional Algorithm system in place and operational
Success End Condition	User uploads images & Correct data is returned
Failed End Condition	Image upload/processing Error
Primary Actors, Secondary Actors	User, FrontEnd system, Lambda Function
Trigger	User uploads image
DESCRIPTION	
Step	**Action**
1	User navigates to page
2	User uploads image
3	Data is returned to the user on the page
EXTENSIONS	
Step	**Branching Action**
3a	User uploads another image
VARIATIONS	
Step	**Branching Action**
2a	User recieves an error



2.4 Constraints

Time

- **Description -** This project must be completed by the end of the Dublin City University 4th year, deadline 19 May 2019 23:59

Artwork (Paintings)

- **Description -** The domain of this project has been chosen as to allow for difficulty but was reduced from all images to images of artworks, specifically painting because of the time frame and the complexity with regard to identifying all images

Functional Constraints

- **Description -** The project should include all the functionality that this document describes, and it should be obvious that the functionality exists.

Non Functional Constraints

- **Description -** The project should include all the non functionality, but yet critical aspects that this document describes, and it should be obvious that the needs of the system have been met.

User constraints

- **Description -** The features should be easy to use or quick to learn for a user of web browsing level knowledge. It should be intuitive to pick up and should not require memorisation to be able to navigate well.

3. Functional Requirements

- **Description -** Must correctly identify image as artwork
- **Criticality -** Moderate, Without this the database will fill with irrelevant info but this feature can be switched off to use a static database
- **Technical issues -** This is a bleeding edge area of research. This will include many hard to solve problems including facial recognition and object recognition
- **Dependencies with other requirements -** This depends on the web scraper finding images and being fully operational.

- **Description -** Must identify artwork as similar despite being slightly altered
- **Criticality -** Critical, Without this the program does less than a hash of the image. The system needs to have a robustness about it as to not be fooled easily.
- **Technical issues -** This will be difficult as there is a lot of research still going on in the area of image vision and depending on where we draw the line finding key points in the images can remain a difficult task
- **Dependencies with other requirements -** This depends on the user interface delivering an image to the system and the data extraction algorithm working well.

- **Description -** Must be able to return a result within a defined reasonable time period
- **Criticality -** Moderate, This system is designed to be fast and usable by a day to day user. If the wait is beyond 10 seconds the user interface is not very useful and the project will be more of a proof of concept than a usable day to day product.
- **Technical issues -** This will be helped with good architecture and a lot of precomputing of results to be used at a later date.
- **Dependencies with other requirements -** This depends on the user interface delivering an image to the system and the data extraction algorithm working well.

- **Description -** Must be able to find images unsupervised
- **Criticality -** Moderate, This system is designed to have a large range of images from all across the internet. This algorithm should be able to find these images across the internet and place the precomputed data into the database. Without this the database will contain a static amount of data, unless populated by hand

- **Technical issues -** This is not as technically demanding as other requirements although does come with a huge overhead and is not necessarily a small feature by any means
- **Dependencies with other requirements -** This depends on being able to identify an artwork in the image and also being able to write to the database.

- **Description -** Must be able to store metadata about images not the image
- **Criticality -** Critical, Without be
- **Technical issues -** This is not as technically demanding as other requirements although does come with a huge overhead and is not necessarily a small feature by any means
- **Dependencies with other requirements -** This depends on being able to identify an artwork in the image and also being able to write to the database.

3.1 Non Functional Requirements

Time

- **Description -** This project must be completed by the end of the Dublin City University 4th year, deadline 19 May 2019 23:59

Speed

- **Description -** The user is in the context of a fast paced world and will only wait for a short period of time for the results to be returned to them. It will therefore be a constraint to get this system to be able to return an accurate result in under 10 seconds.

Ease of use

- **Description -** This system should be a simple to use tool aimed at all levels of tech to be able to use the system. The system should take from other search engines the simplistic and intuitive designs that most come with in the modern day.

Storage

- **Description -** This project may require vast amounts of storage to be able to store a database of considerable size. Seeing as we will be using an Amazon S3 instance to store our data with will not be a constraint, but it will add to the cost constraint. This storage solution also needs to be fast to suit the speed requirements and I think this professional solution from Amazon will fit that requirement

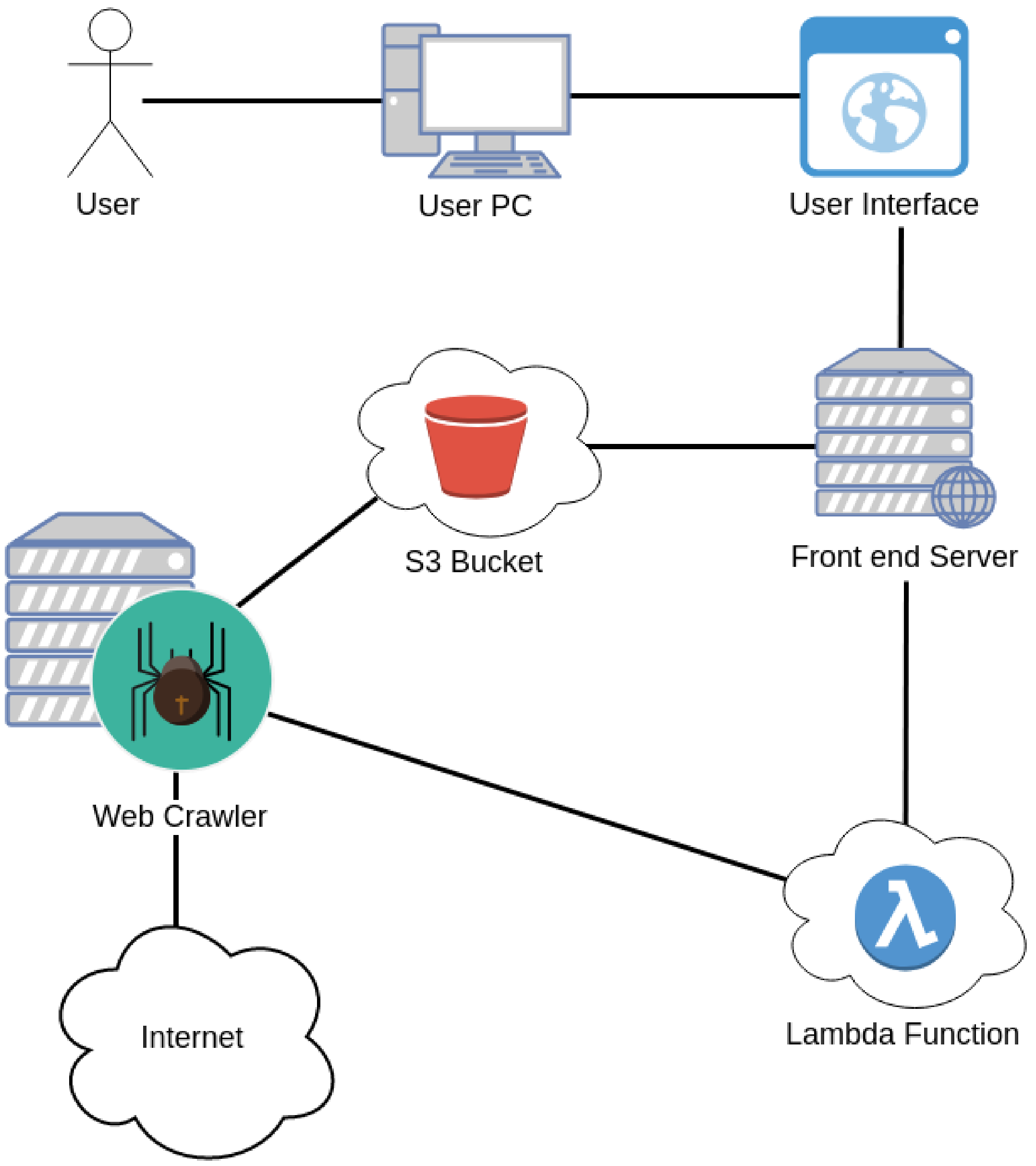
Costs

- **Description -** This project does not have an infinite budget and therefore it has been decided to use Lambda functions from Amazon. This allows there server that has to run constantly to be very small and require very little management as it will not be doing any of the image processing keeping the day to day use of the server very low and therefore cheap. But should a request require image processing it will be handed to a Lambda function, a serverless solution to running code.

Hardware

- **Description -** Image processing takes vast amount of processing power to be able to do in a reasonable time and hardware can cost a significant amount of money raising costs of this project. This constraint means that we are using small hardware where possible and Lambda functions where heavy lifting is needed. Meaning we do not need any physical servers or hardware to run our project on it will all be operated in the cloud as serverless infrastructure for the most part.

4. System Architecture



5. High-Level Design

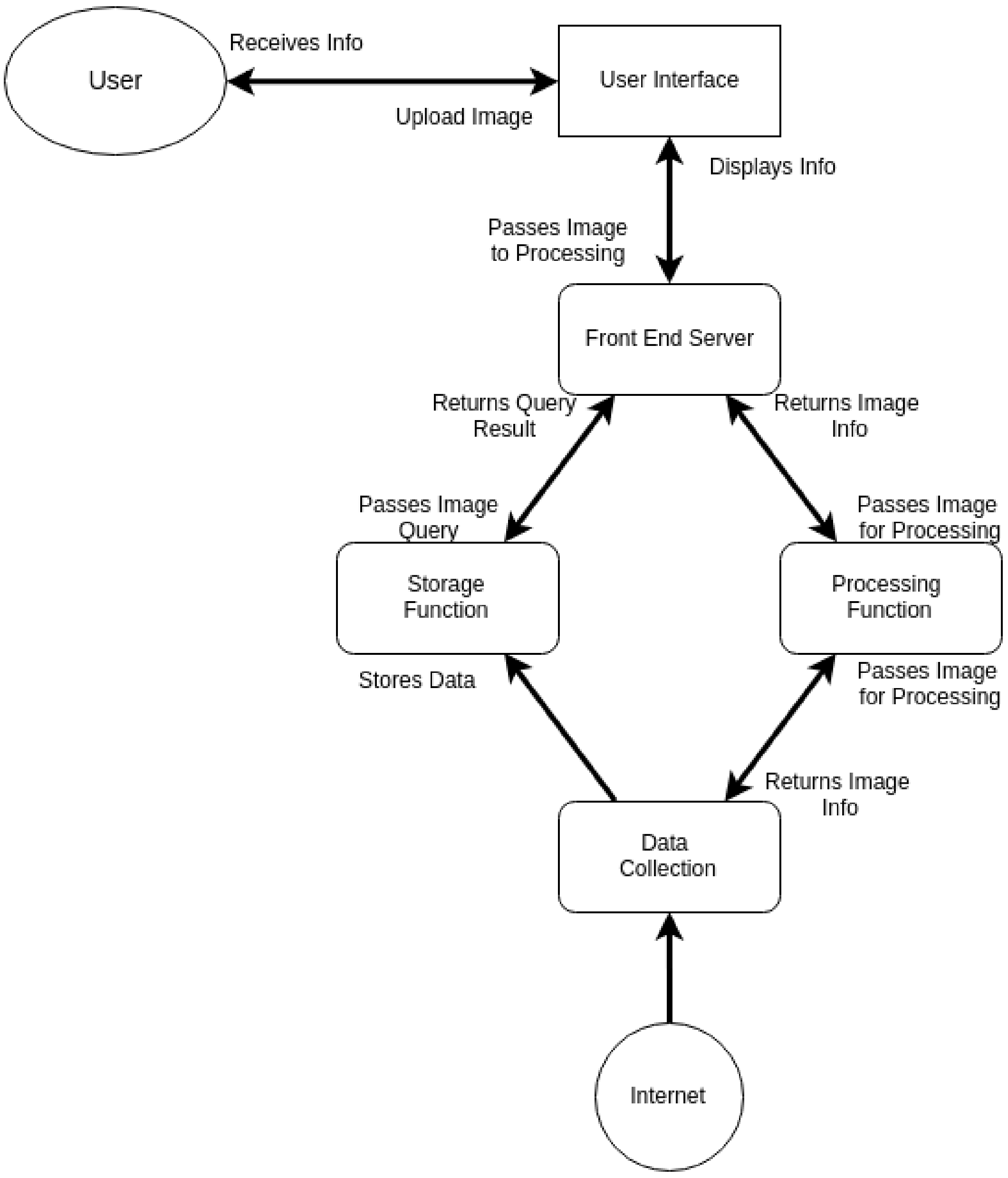
The design of the project is to use only the resources that are necessary and to scale as needed in a elastic fashion. One of the ways to do this is to design with computationally difficult things only having the resources to run when they are needed. This can be done but utilizing new and bleeding edge serverless cloud solutions, such as lambda functions.

These new cloud computing resources allow you to call a container with just a small amount of code and that container will run that code and return a result. The container only exists for as long as the code takes to run, therefore you only pay for the time that the container is alive saving resources and a by product, money.

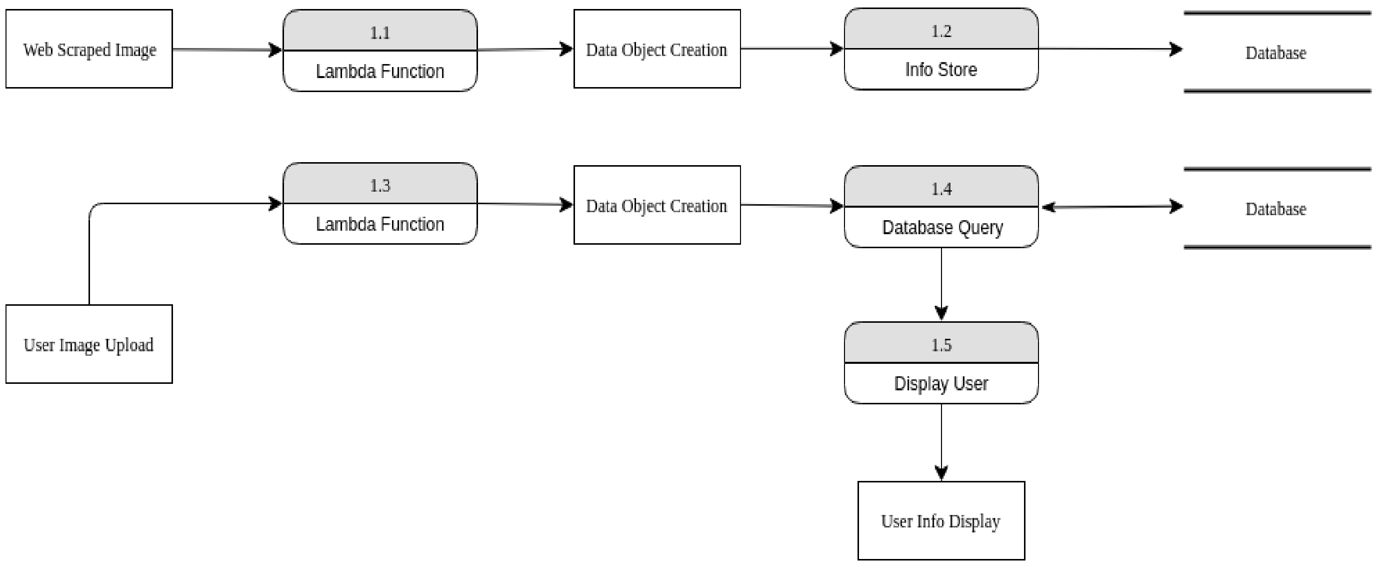
The plan for this system is to utilize this new serverless approach to do the heavy lifting in the project which is the image processing aspect of the search. What this does for the design is takes the heavy pressure off the web server machine and allows that to route traffic as normal, also giving the users no sluggish feeling or slow responses. This also allows for scale as the more users that come to the web application the service does not slow down because of many many users coming to the service. It merely is serving text and all the processing is not done on the machine.

The other cloud aspect of the project is that Amazon S3 object datastore. This is a highly available and reliable datastore that is managed by Amazon. This will allow fast write and reads and also allow all the data to be stored on a seperate machine to the web server and the web scraper. This also improves performance because datastores, especially one that is constantly scraping and writing to disk are computationally expensive. There is be a latency with writing and retrieving data because of the cloud nature that is has to read from a remote location, but it is believed that this impact does not out weight the benefits of separating them.

The web server and the web crawler will live on the same physical machine but because they are containerised away from each other they will be treated for the rest of the project as two separate systems because of the ease of migrating them, they very well could be move to separate physical systems and they should not care.


User
Receives Info
Upload Image
User Interface
Displays Info
Passes Image
to Processing
Front End Server
Returns Query
Result
Returns Image
Info
Passes Image
Query
Passes Image
for Processing
Storage
Function
Processing
Function
Stores Data
Passes Image
for Processing
Returns Image
Info
Data
Collection
Internet

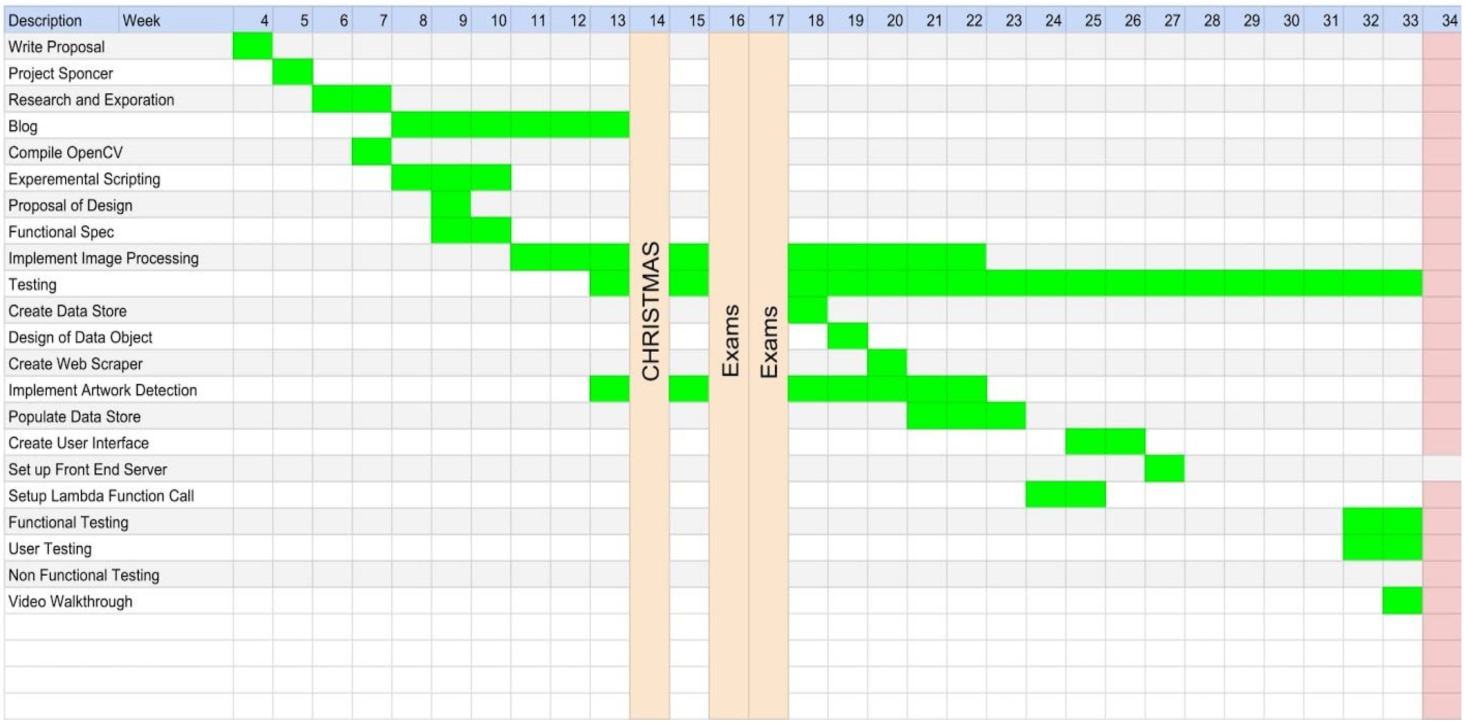
5.1 Data Flow Diagram



6. Preliminary Schedule

The following is a rough schedule of start and stop times for the major components of the project. As college projects should, this will push the limits of my knowledge and as such there will be a lot of push back because of learning curves of new technology and research. I can see this interfering heavily with the schedule and so I have left a grace period of a few weeks near the end to allow myself time to catch up on what ever I have found has been slower than expected.

There is also a lot of deployment configurations in this project and without ever having done anything like this before I am finding it hard to estimate how long it will take me. Despite all these setbacks I feel this is a very good preliminary indication of the life of the project for the next 34 weeks.



Description	Week	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34
Write Proposal		🟩																														
Project Sponcer			🟩																													
Research and Exporation				🟩	🟩																											
Blog						🟩	🟩	🟩	🟩	🟩	🟩																					
Compile OpenCV					🟩																											
Experemental Scripting						🟩	🟩	🟩																								
Proposal of Design							🟩																									
Functional Spec							🟩	🟩																								
Implement Image Processing									🟩	🟩	🟩		🟩			🟩	🟩	🟩	🟩	🟩												
Testing											🟩		🟩			🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	
Create Data Store												CHRISTMAS		Exams	Exams	🟩																
Design of Data Object																	🟩															
Create Web Scraper																		🟩														
Implement Artwork Detection											🟩		🟩			🟩	🟩	🟩	🟩	🟩												
Populate Data Store																			🟩	🟩	🟩											
Create User Interface																							🟩	🟩								
Set up Front End Server																									🟩							
Setup Lambda Function Call																						🟩	🟩									
Functional Testing																														🟩	🟩	
User Testing																														🟩	🟩	
Non Functional Testing																																
Video Walkthrough																															🟩	

7. Appendices

The Public Domain vs. the Museum: The Limits of Copyright and Reproductions of Two-dimensional Works of Art
(https://www.jcms-journal.com/articles/10.5334/jcms.1021217/)

National Portrait Gallery and Wikimedia Foundation copyright dispute
(https://en.wikipedia.org/wiki/National_Portrait_Gallery_and_Wikimedia_Foundation_copyright_dispute)

Copyright in Photographs of Works of Art
(http://www.museumscopyright.org.uk/resources/articles/bridgeman/)