**Automatic Multimedia Metadata Tagging & Text Summarization**

**A Proposal Submitted By**

Group 2

December 4th, 2022

Eli Cohen: elicohen2018@fau.edu

Ryan Smith: ryansmith2018@fau.edu

Jose Silverio: jsilverio2018@fau.edu

Giovanni Gonzalez: ggonzalez2019@fau.edu

Kyle Mendoza: kmendoza2018@fau.edu

**Advisor**

Hari Kalva

Hkalva@fau.edu

**Sponsor**

US Navy (NETC)

randall.l.denney.ctr@us.navy.mil

**Project Summary**

The project objective is to create a system in which files can be accessed readily and quickly by service members. An educational file is uploaded, and the file is read, a summary is created based on the information, context tags, and metadata, and the generated data is uploaded to a database to be searched.

[**1 Introduction**](#_n1b7wol2keb5) **3**

[1.1 Background](#_67w0hgyafweb) 3

[1.2 Statement of the Problem](#_vztayfdwcet1) 3

[**2 Scope of Work**](#_2m0voykrniu4) **4**

[2.1 Overview](#_86z4pm2n269a) 4

[2.2 Literature Review](#_55e63pgqz2r5) 5

[2.3 Alternative Solutions](#_mhc05z2957c7) 7

[2.4 Evaluation](#_s48qkl6r96a6) 7

[2.5 Decision](#_4nl315x04iz3) 7

[**3 Implementation Details**](#_2wom11jlayv1) **8**

[3.1 System Specifications and Functionalities](#_ckd57n1r4jp2) 8

[3.2 Overall System Design with Block Diagrams](#_li7d5hvyqjzr) 9

[3.3 Circuit diagrams, Flowcharts, and/or Use Case Diagrams](#_ipzcjmvxebay) 10

[3.4 Testing Results](#_uhfazouckvte) 15

[3.5 Discussion on Lessons Learned](#_mgsft4wz4mm) 20

[**4 References & Technologies**](#_9601iw5cw2ax) **20**

[4.1 Programming Languages](#_i584tdwxxrg2) 20

[4.2 Technologies & Services](#_ttpgi2cs0q1a) 20

[4.3 Software](#_j64w1ku4ooix) 20

[4.4 Libraries](#_bkujsodah8bw) 21

[**5 Conclusion**](#_j7mx9d3azzyz) **21**

[**6 Personnel/Organization Chart**](#_gc257al50ro1) **21**

# 1 Introduction

The purpose of this paper is to cover the design and development of a software development project with the purpose of creating a solution that will solve some of the problems presented by the sponsor. The Sponsor NETC is interested in developing a system that allows the tagging of educational content in different aspects, such as automating metadata tagging and researching image and figure detection in a document such as a PDF. The overall interests of NETC are educational training and creating a system that can reduce training time and costs.

## 1.1 Background

The purpose of this section is to detail the purpose and reason for the design and development of this project. The US Navy Training Command Center (NETC) reached out to our team at Florida Atlantic University as part of the course for young engineers to create projects with real-life applications. The problem proposed is as NETC follows: The United States Navy has many training documents, videos, and picture files. When students are studying, it is hard for them to go through all the Navy documents to find the required information. This project will split out the training resources into sections, create a metadata tag and a summary, and insert this information into a database from different subject areas that can be easily searched for. The database will also hold a link to the original document and automatically go to the page number or point in the video where that information was retrieved from. This problem is being considered because, currently, NETC does not have an effective system to disseminate and make use of the training documents. The problem proposed by NETC is very important for reducing training time and cost and improving mission readiness. NETC has proposed this problem to a previous FAU team, and they have created a website that lacks functionality and does not touch upon the importance of the project. Our goal is to improve and redesign our solution to solve this shortcoming of the previous groups.

## 1.2 Statement of the Problem

The problem is that the US Navy currently does not have a way to educate service members in an active environment. This means that education mostly takes place in a classroom. However, due to a large amount of existing information, it’s common for an individual to lack knowledge of certain criteria. The only solution is to notify a more experienced individual, which is not ideal in critical moments. As such, the objective of this project is to create a system in which educational multimedia files can be accessed readily and quickly by any service member, advancing the learning process. The benefits of such a system are everywhere and can be used in a multitude of applications ranging from the educational field to the medical field to even business applications. The only current solutions available to this problem are previous attempts from previous university groups and separate solutions for different components of the overall project, but no unified system as described in the problem description.

# 2 Scope of Work

## 2.1 Overview

The main solution of this project required reaching certain milestones regarding different stages of development starting from back-end to front-end such as:

1. Project Research

a. Understanding The problem

b. Determining the Scope of the Project

c. Functional Requirements of the Project

d. Creation of the Preliminary Project Proposal

1. UI/UX design
2. Diagrams
3. Use cases
4. Proposed Structure

e. Creation Subsystems & Personal Roles

1. Subsystems
2. Metadata, Text Summarization, & Keywords
3. Image Extraction
4. Video Parsing
5. Database management
6. Server Management
7. Website Designs
   1. Roles
8. Team Lead
9. Back-End Developers
10. Front-end Developers
11. Management Server

f. Research into How individual subsystems

g. Creation of Main Implementation report

1. Detailing how the implementation of each subsystem
2. Diagrams
3. Wireframes
4. Specifications of project requirements
5. Determining Software and Technologies
6. Project Schedule Plan

2. Project Implementation

a. Personal Development of each subsystem

1 Technology associated with each subsystem

2. Diagrams & Use case diagrams

b. Documentation of the Development of each subsystem

c. Following of Project Planner

d. Integration of each subsystem

e. Testing of Subsystems

f. Weekly meeting with Sponsors

1. Updating Sponsors on progress

g. Implementation of Live Server

h. Testing phases for integrated subsystem

i. Project Scope Review

j. Completion of Project

3. Creation of Project Press Release

4. Creation of Project Poster

5. Creation of Final Report

a. Detailing the current project

b. Lessons learned

6. Creation of a GitHub to contain Project files

7. Creation of Final Deliverable File

1. src (source code for your project, build instructions, Create sub-directories as needed.)
2. doc (project report, flier, research material, any other documentation)
3. photos and videos (demos, presentations, videos)
4. data (any data sets used in your project; add a document describing the data sets)
5. other (folder for any other contents)
6. contents.txt (describe contents of each of the sub-folders)
7. build.txt or build.docx (instructions on how to build and reproduce your app/service/project)
8. test.txt or test.docx (instructions on how to test your app/service/project)

## 2.2 Literature Review

To prepare for the project, we researched patents with similar findings and topics to our project. We needed to understand the progression behind other researchers and the discoveries they made throughout the process. Below is a list of patents we found with somewhat related subjects to our project.

Patent 1

Patent number: US8935204B2

Title: Metadata tagging of moving and still image content

Inventor: David Peto, Stef Lewandowski

Summary

This patent is for the method of image content with metadata tagging through the use of voice recognition software. The user speaks using their voice, which identifies keywords throughout the audio stream and the output metadata is synchronized with the relevant contents of imagery. Using the control console, a human tagger is capable of navigating onscreen menus and choosing from different taglines, providing multilevel tagging of metadata based on the image content. The stored metadata is inside an integrated system full of tagged digital image content.

Relevance

The scope of this patent is similar to our project in the aspect of metadata extraction. While this patent uses voice input for selecting metadata taglines, our project uses a keyboard that types up an algorithm for metadata extraction from files. After further research into this patent, the invention relies on a human tagger which is the purpose of the voice recognition software, using the audio input as a way to generate tags. However, understanding the method behind the process of metadata extraction, this helped the group research a proper algorithm to extract metadata from files and find a storage destination.

Patent 2

Patent number: US8370358B2

Title: Tagging content with metadata pre-filtered by context

Inventor: Jyn-Han Lin, Vanessa Tieh-Su Wu

Summary

This patent is for the generation of tags from metadata using context. In a plurality of data items, there is metadata present. After the context is determined for a user, a data item is selected and the received content is used to compare with the selected data items as a way to identify a similarity. The purpose of this patent is to simplify the problem of manually tag content on computing devices by giving an automated solution. In this case, tag generation no longer requires user input and an algorithm is used to cross reference tags and sort them by similarity depending on what is being extracted.

Relevance

The scope of this patent is similar to our project due to the requirement of automated metadata extraction. Our goal is to move away from manual input of metadata in order to simplify the process of service members quickly having submitted files identified and produced with the right tags. This will help speed up the process as time is a valuable resource when they are out in the field. In this patent, metadata is matched through a location of the computing device, which is where the content is captured, as well as using pre-existing structured data acquired by the user, which is also true in our project when a user submits a file to the website, it is processed.

## 2.3 Alternative Solutions

Method 1: Updating an existing website

This method involves updating and improving the original website created for this project by a previous research team. This would have involved using the HPC server provided by FAU as well as a PHP connection instead of the AWS approach that we decided to do. This has the benefit of using an existing framework and building off of something that was already created in order to help the team save time during the development process. The group would have determined the subsystem requiring the most focus and diverted the majority of available resources toward its progression.

Method 2: Foregoing Front-End

This method would forgo the front end section and remove the server component and focus on creating a program that would allow the user to run large batch files through the system and create a directory with the newly generated content. Then we could link a database and SQL queries to look for content associated with metadata, keywords, and by doing so we can create an offline system that allows for an integration at a later time with some type GUI interface.

Method 3: Building Front-End + Back-End

This method would involve starting from scratch, establishing a server and database using AWS, creating file storage through the S3 bucket, and using Bootstrap with CSS to customize the web page appearance. MySQL workbench would be used to write SQL queries that will communicate to the Amazon RDS and have the ability to fetch data from the database. Flask will be used as a routing service for file access. In addition, Python will be used for image extraction and text summarization.

## 2.4 Evaluation

The primary focus of our website is complexity, available resources, and time constraints. The website’s ability to create a storage destination through the S3 bucket is the most important because it serves as an endpoint for all data. In terms of complexity, this was low on the list because AWS offers this service with the classification of our choice. This is directly related to time constraints because the development time was quick because the majority were preconfiguring settings established by Amazon. In terms of available resources, we were provided 5 GB for file storage. Our team was fairly new to database management, so we focused on subsystems that took less time to develop due to background experience. Having this mindset, we started with front-end development and website creation, followed by the creation of the S3 bucket. Next, we focused on image extraction, keyword extraction, and metadata, and lastly, the database.

## 2.5 Decision

Based on all the methods presented we chose to implement method 3 for a myriad of reasons. The reason method 3 was chosen was due to the fact that it is a current and modernly used industry standard. This is important for the longevity of the project and maintenance of the software. Secondly, the scalability of using method 3 allows the creation of a more specified project that allows the development of a specified project tailored to requirements of the problem. Thirdly, by building the project from the ground up the developer can insure what goes in and what goes out. Lastly, we chose the third option due to the fast development cycle and using AWS to host our website.

# 3 Implementation Details

## 3.1 System Specifications and Functionalities

The following section covers overall project requirements that will dictate what the solution will do regarding functionality and usability.

Functional Requirements:

F.1 Extract Metadata

1. System can extract metadata from various file formats
2. System should send extracted metadata to a storage system

F.2 File Uploading

1. System can recognize MP4

B. System can recognize PDF

F.3 Image/Figure Extraction

1. System can extract images/figures from file formats
2. System can extract Image page by page
3. System can send Images to the file storage system

F.4 Keyword Extraction

1. System takes extracted text summary
2. System extracts keywords from the text summary
3. System sends extracted keywords to database

F.5 Website Design

1. Landing page

B. Content page

C. Search bar

D. PDF upload button with upload page

E. Video upload button with upload page

Usability Requirements:

U.1 Limited file formats supported (.pdf and .mp4)

1. Users can only upload certain types of files

U.2 Processed Data

1. Data that has been summarized and other processes displayed in a standardized way

Requirements not met:

Functional:

F.1 Audio and Video Transcription

1. System can transcribe audio and video files
2. System can create timestamps for transcribed section

F.2 Search Engine

1. System must be able to search the database
2. System must be able to produce results searched keyword

F.3 Content Control

1. System allows user to add keywords
2. System allows user to add and delete content

F.4 Login

1. System allows for content editing through a user account

Security:

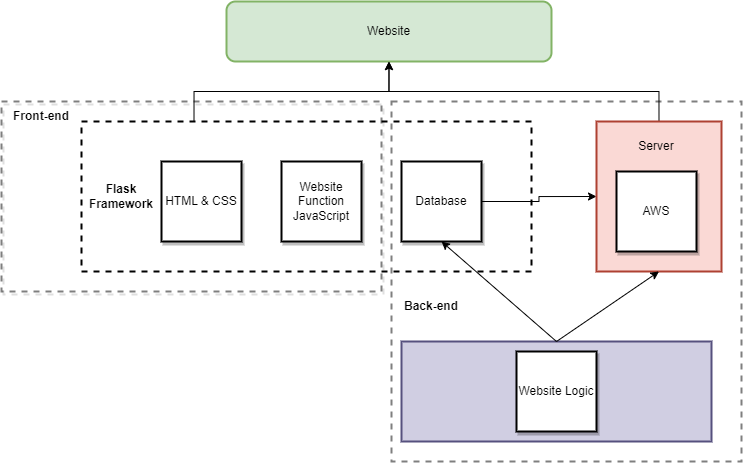
S.1 Multimedia Encryption of files

1. Adaptive File Encryption

These requirements were not met due to failed attempts in implementing each one. As such, they were dropped to conserve time and focus on sub-system implementation.

## 3.2 Overall System Design with Block Diagrams

The product description is as such, the user accesses the website and is presented with the option to upload various multimedia documents. A document is uploaded, and the website processes this document in multiple ways. First the document is saved in the database as the original, then the document metadata is read, and tags are generated based on the content in the document. After tags are generated a text summary and image extractor is performed and linked to the original document. After the data is stored and processed, the processed data is called and presented to the user in an informative manner.



**Figure 1: User Access Description of The Project**

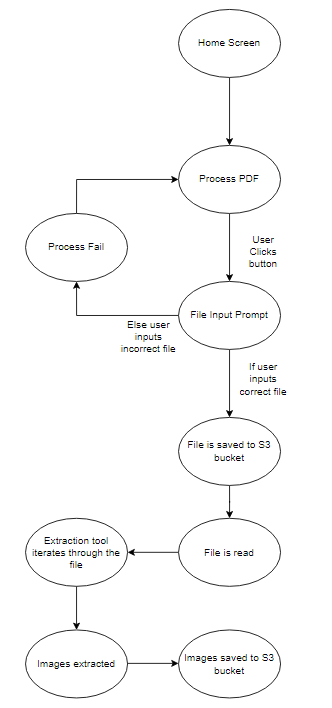
In Figure 1, what is represented is the overall segmentation of the website into key sections of interest and how each of these sections should interact with each other. To briefly explain, there are two main sections in software development which are front-end and back-end. The two main sections are broken down into further sections that include different subsystems. Each of the sub-systems are broad and complicated and dictate a necessity of various talents to create a cohesive project. The sections and subsystems can be explained as such, the front-end side is responsible for the visual information, user functionality, and incorporates the Back-end functionality to the website. The back end is responsible for many of the inner workings of a website such as information storage, security, and the overall website logic that allows users to use functions that are not evident. The back-end also allows for server integration functionality towards the website.

## 3.3 Circuit diagrams, Flowcharts, and/or Use Case Diagrams

**Extracting Images from PDF Documents**

(Team Members Responsible: Ryan Smith)

Ryan Smith handled the extraction of images from PDF documents. I utilized and added the functionality to the buttons for processing PDFs and extracting the images from the uploaded files. The processing PDF or video buttons will allow the user to click and have a pop-up appear, allowing the user to upload a valid PDF or video. If the user submits something correctly, then a success prompt, as well as the contents uploaded will be displayed to the user. Furthermore, the uploaded PDF files are uploaded to our AWS S3 bucket. The stretch goal assigned to me was to extract text from the extracted images, although this subsystem works locally, I was unable to make this functional on our website version due to the module incompatibilities with our Linux based Ubuntu server. This system was intended to extract the text from images and save that content to a text file within the S3 bucket.



**Figure 2: PDF Image Extraction**

In Figure 2, the user will be greeted with the home screen of our project and presented with an option to process a PDF. The user can click on the upload PDF button and if the correct file type is uploaded, then the file will be uploaded to the S3 bucket, and if an incorrect file is uploaded then the process will fail and prompt the user to upload another file. The file will then be read with the image extraction algorithm. If the algorithm identifies an image, it will then be saved to the S3 bucket.

**Metadata, Text Summarization, & Keyword Extraction**

(Team Members Responsible: Eli Cohen)

Eli Cohen will create and develop a program that can read text from a file and create a summary of the contents of the text as well as identify key words based on frequency. I plan to read a PDF file and extract text from the file. This text will run through a tool that will create a summarization tool in which. I will implement another tool that will read the summary text and identify key words. The Idea behind the process is as follows: the text summarizer will create a summary for each page and the keyword identifier will read each page summary and produce the keyword for each page summary.

Diagram, schematic

Description automatically generated

**Figure 3: Text Summarization and Keyword Extraction**

In this figure what is represented is the process in which the text summarization, keyword identifier, and metadata extraction will occur. The process begins with the user uploading a file that is the correct format. Then the three things happen: metadata is extracted, the text in the file is read and extracted which is then sent to the keyword identifier tool which generates keywords from the summary. Once these three products are produced it is sent to the S3 Bucket.

**Website Design**

(Team Members Responsible: Giovanni Gonzalez)

Giovanni Gonzalez will handle complete website aesthetics compiled from respective HTML and CSS files. Bootstrap will be used as a framework due to its high compatibility to work in conjunction with HTML/CSS modifications for adequate user interface design. The main elements on the website will be the project’s required credentials, a navbar, and buttons for respective input types (search bar user-text/file input). Project credentials will include necessary founder info such as names of the creators, project repository, project summary, and respective sponsors including contact info. The navbar will be designed in a user-friendly manner for automatic transitions between website features (buttons).

Graphical user interface

Description automatically generated with medium confidence

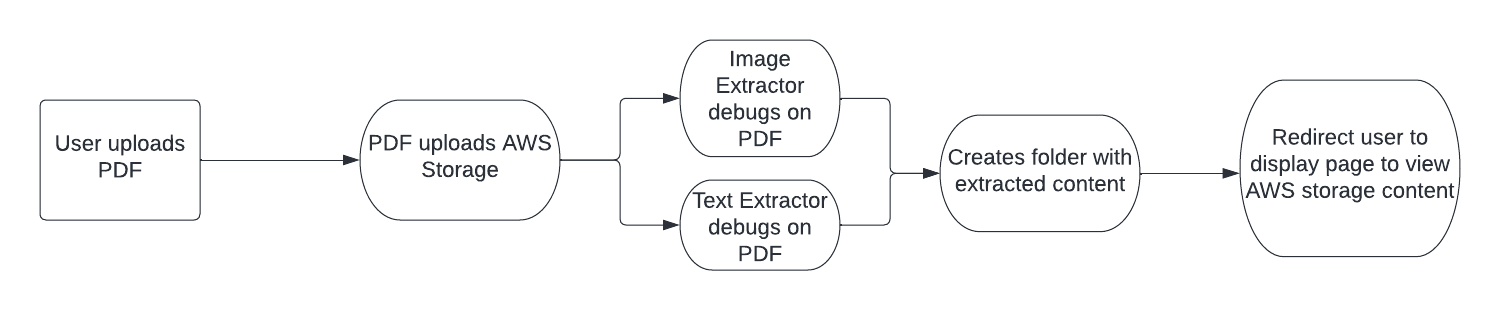
**Figure 4 – Website Design (Front end)**

The state diagram above displays the website design of our project and begins with the user entering the website. The user can click on the navigation bar located at the top of the webpage and will expand to the display. The navbar will have two options to move to the search bar element or doc/pdf and video button uploads. If the user does not wish to use the navbar, they can simply scroll down the webpage to the search bar element and file upload buttons. A pop-up page will be displayed when the user inputs characters into the search bar. Similarly, the user can click on the file upload buttons to upload either a doc/pdf or video.

**Server and Server Management**

(Team Member Responsible: Jose Silverio)

Jose Silverio will be taking care of the server files and the server management which includes the flash application and the server storage. For the server, we will be using AWS for the server management. In AWS, I implemented two systems which are the EC2 instance and the S3 bucket. The EC2 instance will be creating the ubuntu server which will make the ip address live. The S3 bucket will be storing the code files for the web page and also the storage for the documents that will be uploaded from the server. Both these systems will be managed by Jose Silverio.



**Figure 5 - Server Storage Process**

The above diagram covers the process of uploading a pdf to the webpage and going through the server storage. First the user will go to the PDF upload page, it will prompt the user to pick and upload the desired pdf they would like. Once the PDF is uploaded, this will be sent to the server storage which is the S3 bucket. As the PDF is uploading, it will debug both the Image Extractor and Text Extractor. Once the text and images have been extracted, a folder will be created to store the images and texts. Finally, when this process is completed the webpage will redirect to the display page.

**Server & Database**

(Team Member Responsible: Kyle)

Kyle Mendoza will handle the database using AWS and SQL. I will perform data storage and retrieval duties. I will be responsible for providing the pathways for stored data from text summarizations and user submissions. I will also provide a destination for the data coming from the search engines and store it in their proper locations in the database. As the user submits a keyword, this will search the database for a particular pdf, doc, or video of choice and my job is to make sure that the right document is being shown. This database will contain all the stored PDF’s documents, and videos utilized by the website in order to be accessed by the user.

Graphical user interface, application

Description automatically generated

**Figure 6: Server and Database**

In Figure 6, this explains the role of the database on the website. Using Bastion as the server connection, users must log in with the proper credentials and access the contents. Users have the choice to upload a document or video, which will be stored in the database. It will be processed and summarized using the text summarization tool. They can also use a search engine to look for specific keywords in the index, which will go through the database and find the right document or video that is needed. The original document is also stored in the database for future use.

**Table 1**

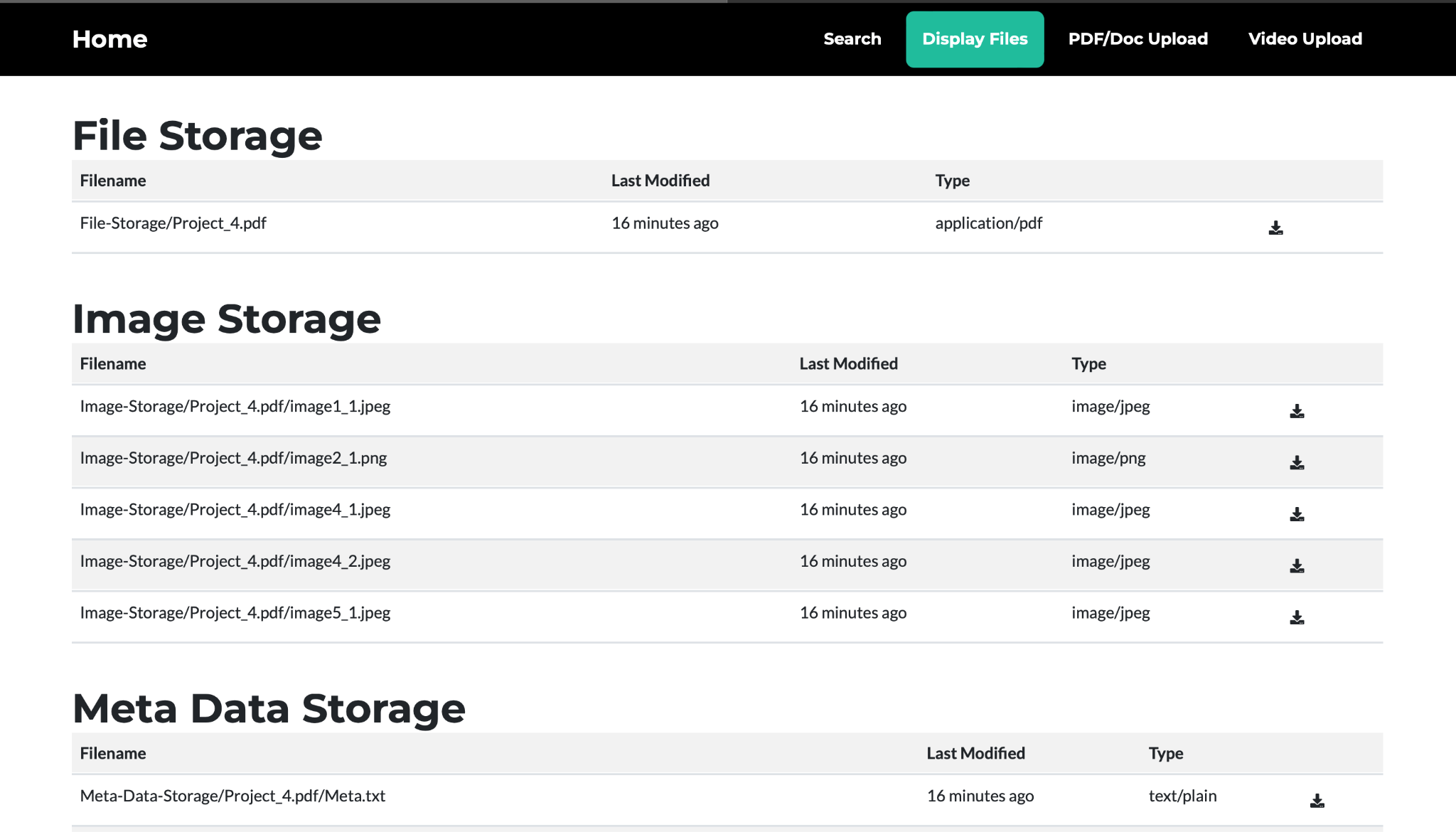
|  |  |  |
| --- | --- | --- |
| **Requirement** | **Subsystem** | **Description** |
| F.1 | Text Summary & Metadata Extraction | Text file is read, and the data is summarized and extracted and is saved to a database |
| F.2 | Image and Figure Extraction | Files uploaded and read by the system and images and figures are extracted from the file saved. |
| F.3 | Keyword Identifier | All processed text from different subsystem is read and keywords are extracted using NLP methods of identifying in which the tags are saved for the specific document |
| F.4 | Web design | The website has a visual representation of the system generated content |
| F.5 | Server / Server Management | The flask integrated server hosts the website and each of the individual webpages |
| F.6 | AWS S3 Bucket | The S3 bucket will hold all files uploaded through the website |

## 3.4 Testing Results



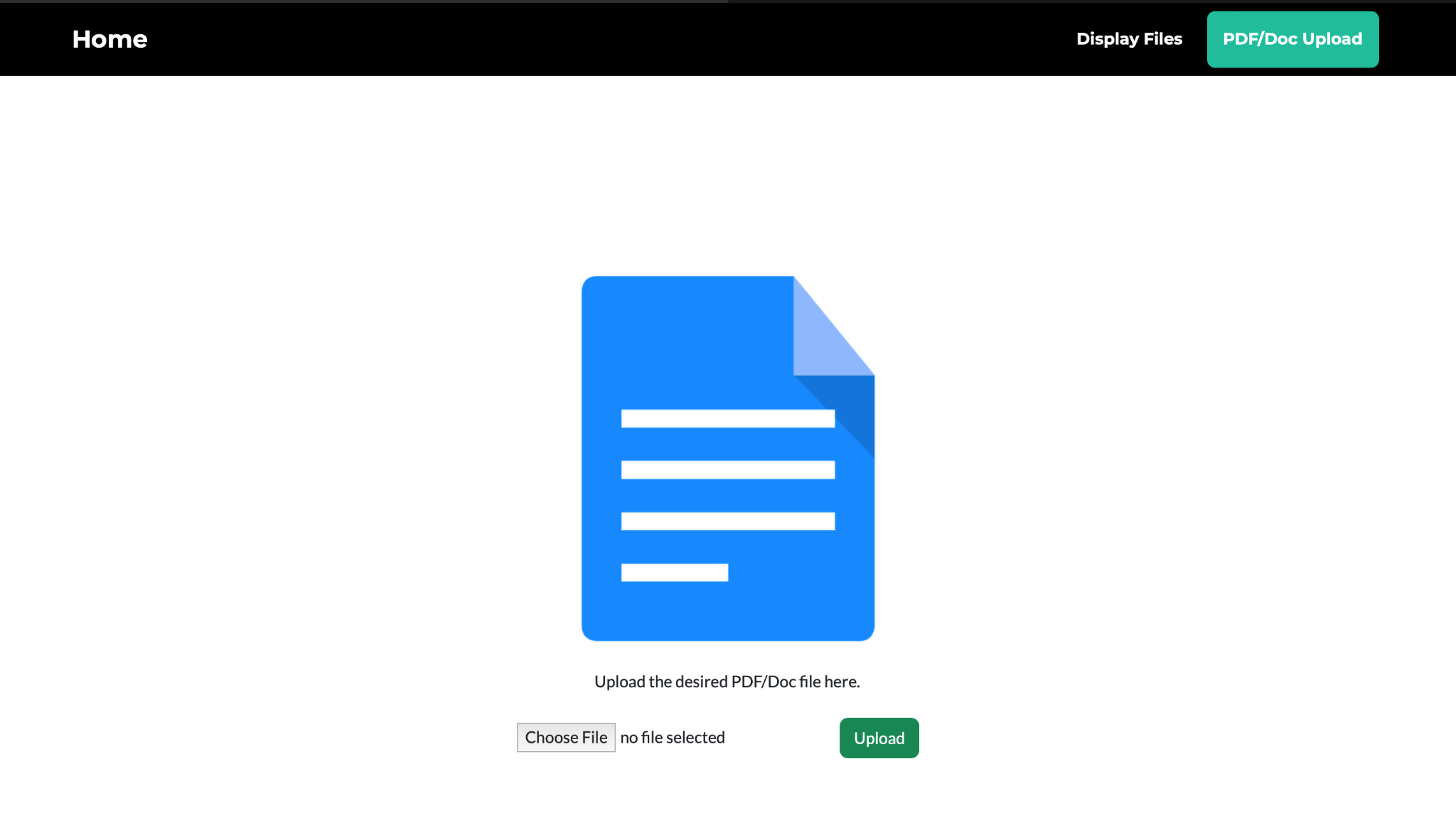
**Figure 7: Home Page**

The home page contains the navigation bar where the user can click on four options to navigate towards the desired location. Two buttons for file uploading are also located in this page. One for pdf format uploading and the other for video. The user can select either button and upload a file of the respective format. The home page also contains general project information and credentials such as descriptions of how the site works, instructions on how to upload, the project repository, and author credits. Testing for this page was carried out by using the buttons to see if the user was correctly re-directed to the respective file upload page (pdf or video).

****

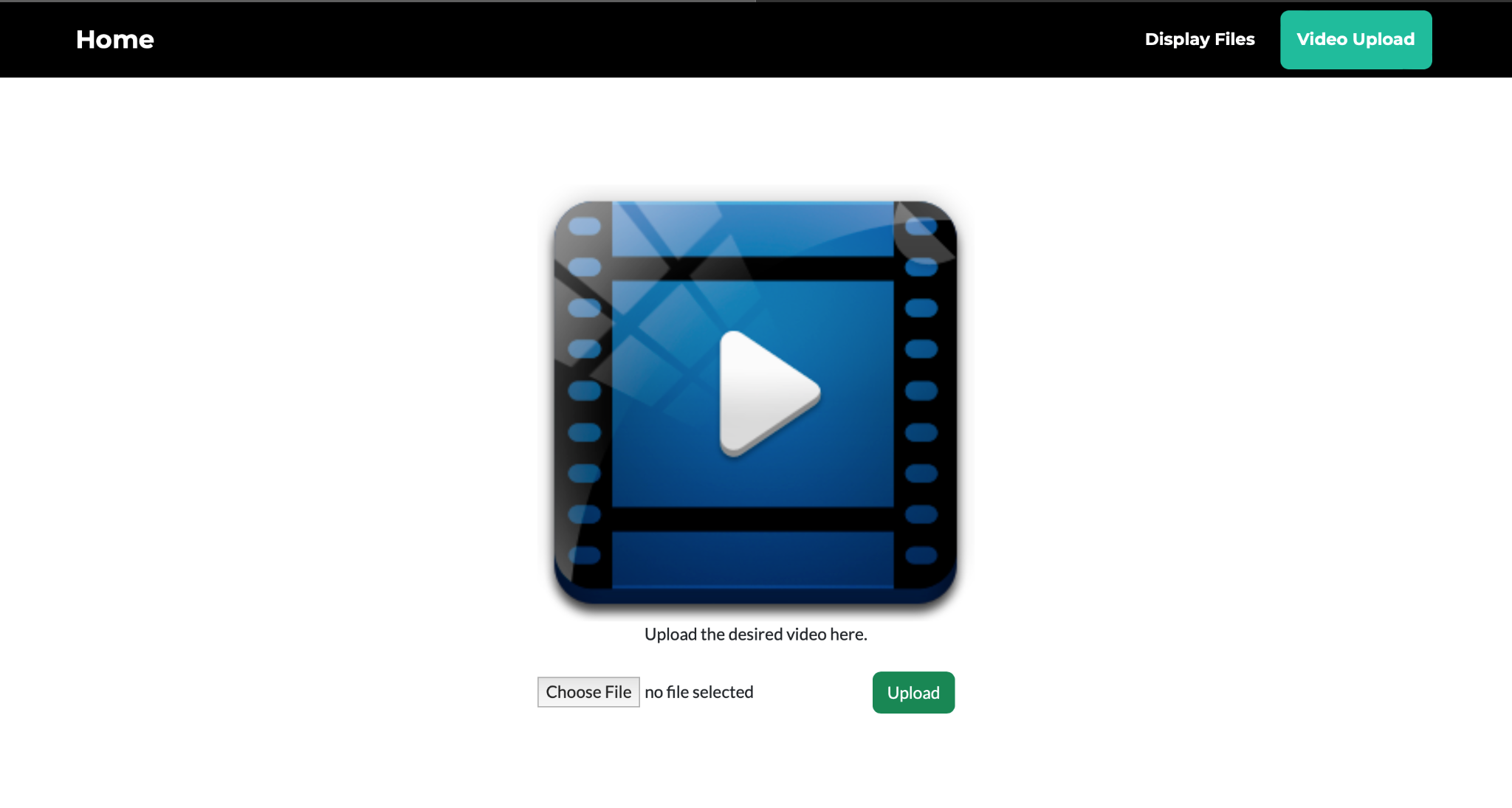
**Figure 8: Display Page**

The display page of our website is used to display the files that have been uploaded to the AWS S3 bucket. As shown in Figure 8, the user can see the items stored within the file storage folder located in the bucket. Inside of the “Project\_4.pdf” folder contains all of the images extracted and are stored within the Image Storage folder. Finally, the metadata extracted from the uploaded pdfs are stored in the Metadata storage folder of the S3 bucket.

****

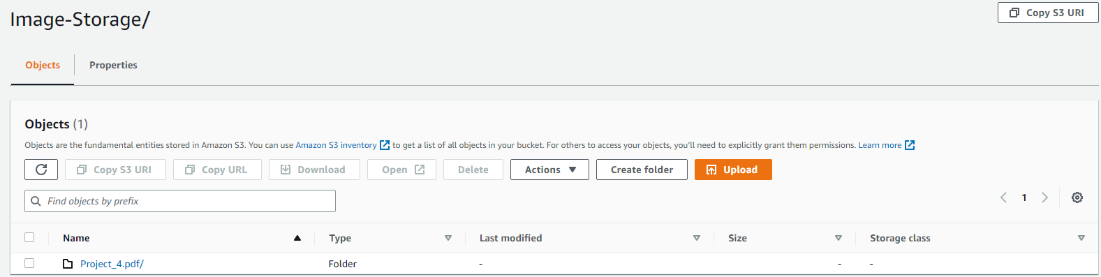
**Figure 9: PDF Upload Page**

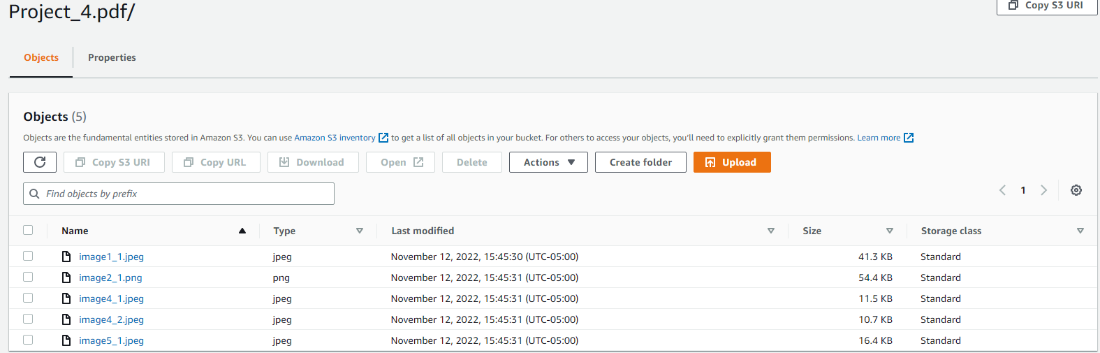
The PDF/Doc upload page of the website allows the user to upload a valid pdc or word document to the AWS S3 bucket. This process will upload the PDF/Doc, but also run the subsystems developed, as previously mentioned, this includes image extraction and text summarization and metadata extraction. When testing the PDF/Doc page, my team continuously modified the code to allow PDFs to be successfully uploaded to the S3 bucket. The user can use the “Choose File” button and brower through their file explorer for a valid file submission. After the user selects a valid file they can click on the “Upload” button, which will automatically upload the chosen file to the S3 bucket. Learning more about AWS was imperative for this section as we needed to understand how to upload files to AWS and make sure that all file types can be accepted into the upload process. This process had us test multiple types of documents or PDFs that the user might attempt to upload.

****

**Figure 10: Video Upload**

The video upload page contains instructions for the user to follow regarding file input. The page contains the function that is responsible for uploading an mp4 file into AWS just like the pdf upload page. Once the user clicks on choose file, file manager opens and permits the user to select the desired file for upload. As long as the video is within a certain size, it will upload to the AWS server. However, the file will not be processed the same as a pdf since it was a stretch goal that we could not complete in time.





**Figure 11: AWS File Storage**

This is the point of view from AWS where the files go once uploaded through the respective buttons. As can be seen, the files are stored categorically with accessible metadata. The example shown is how images are extracted from each pdf. Once a pdf is uploaded, it will be deposited into the bucket and appropriate metadata extracted such as images is deposited into the “Image-Storage” folder. Extensive testing was needed for this section since the file uploading and extracting process was heavily dependent on the routes established by the Flask framework which will be explained below.

Text

Description automatically generated

**Figure 12: Flask Implementation Testing**

The Flask implementation process and testing was imperative for the website as this software was the framework or middle man of the website. Figure 12 displays the routes used within our website's code and shows how our website operates and displays specific pages, such as the home page, labeled as ‘/home’. Furthermore, ‘/home’ is the route that displays the home page of the website. The image above also shows the implementation of uploading files to our file storage system, as we needed to test this functionality thoroughly to allow for the user to upload PDF files to the AWS S3 bucket. Emulating the flask project offline was a large portion of the testing done, and eventually hosted on AWS and tested in a live environment. After various amounts of testing and moving between multiple different pages, as well as uploading multiple different files, we were able to be satisfied with the completion results.

## 

## 

## 3.5 Discussion on Lessons Learned

Throughout the development process of this project, each of us has learned crucial skills for our careers as young Computer Science and Computer Engineering students. To start off, some of the general skills we learned are exclusively related to STEM. Those skills being, how to create a planner, how to document progress, convey ideas, and communicate with a client. These skills are used in everyday life, and in the professional world, these are skills that are developed through real experience. For the second aspect of this project, we learned specific skills related to our respective fields. Such skills include learning Flask, AWS, Linux, Github, and implementing much of the theoretical knowledge we learned in our majors. Overall, this experience has led all of us to understand what our strengths and weaknesses are and what we should focus on as we progress in our field.

# 4 References & Technologies

## 4.1 Programming Languages

HTML and CSS will be the focus point languages for overall website design. These will be implemented in Brackets, which is the chosen code editor due to personal experience and functional comfort since it provides a simple yet efficient developing environment due to its many features and tools for website design. JavaScript will be used in the project to add functionality to the navigation bar created in the HTML of the code. The final programming language used on our website is Python, as it has multiple modules to help implement the subsystems.

## 4.2 Technologies & Services

The technologies and services used in our project include GitHub, Amazon Web Service (AWS), and Flask. GitHub was utilized extensively throughout this project to share the implemented code between all members of the group as well as find resources that could aid in the development of our website. AWS operates as a website hosting service and a storage method through the S3 bucket. Flask is the website framework used throughout our project and allows the linking of Python files.

## 4.3 Software

Visual Studio Code will be used to implement and modify the subsystems of the project, as well as implement these subsystems into our Flask framework. Brackets were utilized by the front-end developers to quickly modify and implement the pages displayed on the website, such as the homepage and the upload files page.

## 4.4 Libraries

The libraries used in the website include:

* Fitz from PyMuPDF – <https://pymupdf.readthedocs.io/en/latest/module.html>
* PyTesseract – <https://pypi.org/project/pytesseract/>
* PDFplumber – <https://pypi.org/project/pdfplumber/0.1.2/>
* Spacy – <https://spacy.io/usage>
* Flask – <https://flask.palletsprojects.com/en/2.2.x/>
* Boto3 – <https://boto3.amazonaws.com/v1/documentation/api/latest/index.html>

# 5 Conclusion

Throughout this project, we have accomplished many feats. We managed to create a website that is hosted on a popular hosting service. We also learned about the software development process, such as creating documentation, project planning, creating a project schedule, and more minor skills like time management and communication. For our project accomplishments to briefly summarize, we have managed to create a Web API where documents are uploaded and we run processes that extract metadata, images, keywords, and video uploads and display content to the user. All of this was done using the Flask framework, which allowed us to develop a website quickly and efficiently. However, if this project could be done again, we would take a different route and use less powerful services and more small and practical ones instead, since the learning curve of some of the things we attempted was high. Secondly, with our current knowledge and our newfound understanding of creating a live website, we could fix many of the problems that arose from a lack of knowledge. Though we have encountered many uphill battles, we have created something from scratch since our predecessors did not have the structure or proper, robust documentation. As such, we had to develop our own designs and create the project from scratch.

# 6 Personnel/Organization Chart

The following organization chart shows the division of labor between team members. Some tasks required a joint pairing due to its difficulty and knowledge requirement to complete it.

|  |  |
| --- | --- |
| **Task** | **Team Member Responsible** |
| Text Summarization Subsystem | Eli Cohen |
| Metadata and Keyword Extraction Subsystems | Eli Cohen |
| PDF Image Extraction Subsystem | Ryan Smith |
| Image-Text Extraction Subsystem | Ryan Smith |
| Website Development | Giovanni Gonzalez |
| File Upload and Testing | Giovanni Gonzalez |
| AWS Server Management | Jose Silverio & Kyle Mendoza |
| Code Integration | Jose Silverio & Kyle Mendoza |
| Flask Implementation | Jose Silverio |
| Database Creation | Kyle Mendoza |

# 