CS4624

Multimedia, Hypertext, and Information Access

May 7, 2022

CS3604 Case Study Library

Virginia Tech, Blacksburg, VA 24061

Instructor: Dr. Edward A. Fox

Client: Dr. Daniel Dunlap

Team: Brian Wieder, Brian Nguyen, Sohil Jain, Kidus Bekele, Kyle Papili, Drew Marin

Table of Contents

Table of Figures

Table of Tables

1. Abstract

CS3604 is a class that explores the context of computing in society through several complex topics. Throughout the semester, students create presentations on specific case studies on topics discussed throughout the semester. These topics include Cyber Ethics, Intellectual Property, Privacy, eCommerce, the Internet (Net Neutrality), and Artificial Intelligence (1). These are hot topics in the field of computer science and interesting to explore more deeply.

For this project, our job was to create a website to house the previous years' case study presentations. To do this we needed to support all of the file types that could be submitted such as PowerPoint, a PDF document, an MP4 video, and a Google Slides link. These files also needed to be categorized to allow for searching based on a variety of categories. These categories include the date, topic, keywords, file format, etc. These files would form a repository of case studies for future CS3604 students to look at as they create their case study presentations. Students can build off these cases or create their own. Students then would submit their case studies directly to the website.

The deliverable for this project is the website itself. The website allows for the submission of a variety of file types and for searching based on the different categories/tags.

The importance of this project is that current students can see the work of students from previous semesters. There is a large repository of presentations that are completely unavailable to students, so this project would allow students to access previous work and build on work across semesters. Another benefit of this website is a streamlined submission process. Students can keep track of all of their submissions for the different sections of the presentation all in one place as well as look at other students' presentations on similar topics and build off of them. This site should also help facilitate a smooth in-class presentation transition. Since all of the presentations are searchable on the website there would be less time spent setting up presentations.

The website was deployed on the VTDLP on the custom domain https://casestudies.cs.vt.edu/. The CS department has funded an AWS account for the website and 638 presentation files have been uploaded to the website. As a team, we tested the website to ensure that a specific presentation could be searched for and that the file was able to be accessed by the user.

2. Introduction

At Virginia Tech, one class required for all computer science students is CS3604: Professionalism in Computing. In general, this semester-long course delves into the field's ethical, social, and professional concerns. It covers the social impact, implications, and effects of computers on society as well as the responsibilities of computer professionals.

A large portion of CS3604 is devoted to case study presentations. These 5-minute presentations are created by each student and cover a wide range of topics, some of which include: Intellectual Property, Privacy, eCommerce, Information and Communication Technologies (ICT / Internet), and Machine Learning/AI/Algorithms (1).

With over 500+ case studies per year, it is Professor Dunlap’s goal to turn these videos into a learning resource for students. The current submission system has proven inefficient and cumbersome to manage, and thus would benefit from a centralized application from which students can access case study materials.

Therefore, the CS3604: Professionalism in Computing Digital Library was introduced as a way to view an archive of student submissions from previous semesters and the ones to come. Not only are students able to easily submit their case study presentations, but anyone who views the library will also be able to search and sort through videos with a variety of filters. The purpose of this project is to allow students to view short, digestible videos/presentations for virtually any topic in computer science.

3. Requirements

Through our conversations with Dr. Dunlap, we established two main goals for the digital library, making this resource as accessible and extensible as possible. For accessibility, there were to be features that would allow the user to search and sort through presentations, making it as easy as possible to find specific studies. Next, making this library extensible meant uploading semesters of previous case studies to the web application. Finally, the digital library should allow various file types and submissions that include large file uploads.

While there were no hard requirements in terms of technologies, it was clear that a web application would best suit the needs of CS3604. The Virginia Tech Digital Libraries Platform was recommended as an option that was already configured and was easy to use by Virginia Tech students (2).

As this is a web application, the user-end requirements are very minimal. The VTDLP front-end is written in REACT.JS. Thus there should be no incompatibility issues on the client side of loading the web application. This was a critical decision as the platform should be easily accessible by all students regardless of their operating system or browser and without the need for users to have multiple libraries pre-installed.

4. Design

4.1 Front-end

At the start of this project, a great resource made available to our disposal was the VTDLP GitHub repository (shown in Figure 1). This platform serves as a bootstrap framework so we can build a customized repository for CS3604 Case Study data.

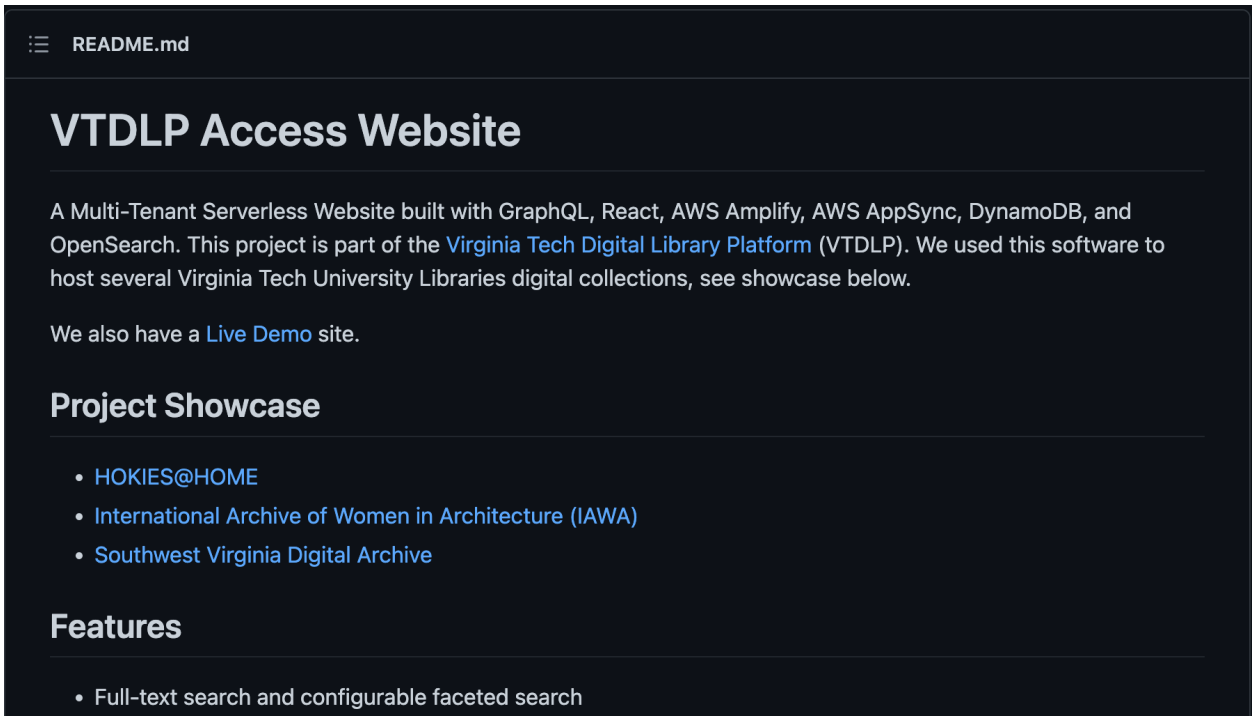

Figure 1: VTDLP Repository

Much of the front-end design work was already completed for this project on account of us making use of the VTDLP access website repository; however, plenty of design considerations remained for us to decide upon. To understand the changes in

design that we need to implement, we must first look at the bare-bones state of the VTDLP front-end. While UI elements are well-rendered and the page is responsive in its design, some elements present in this library are of no use to our specific implementation and will need to be either removed entirely or altered.

Figure 2 is the search/filter page that the VTDLP platform provides for us to make use of. Our web application will need to have search and filter functionality, although not all elements present in this template will be of use to us.

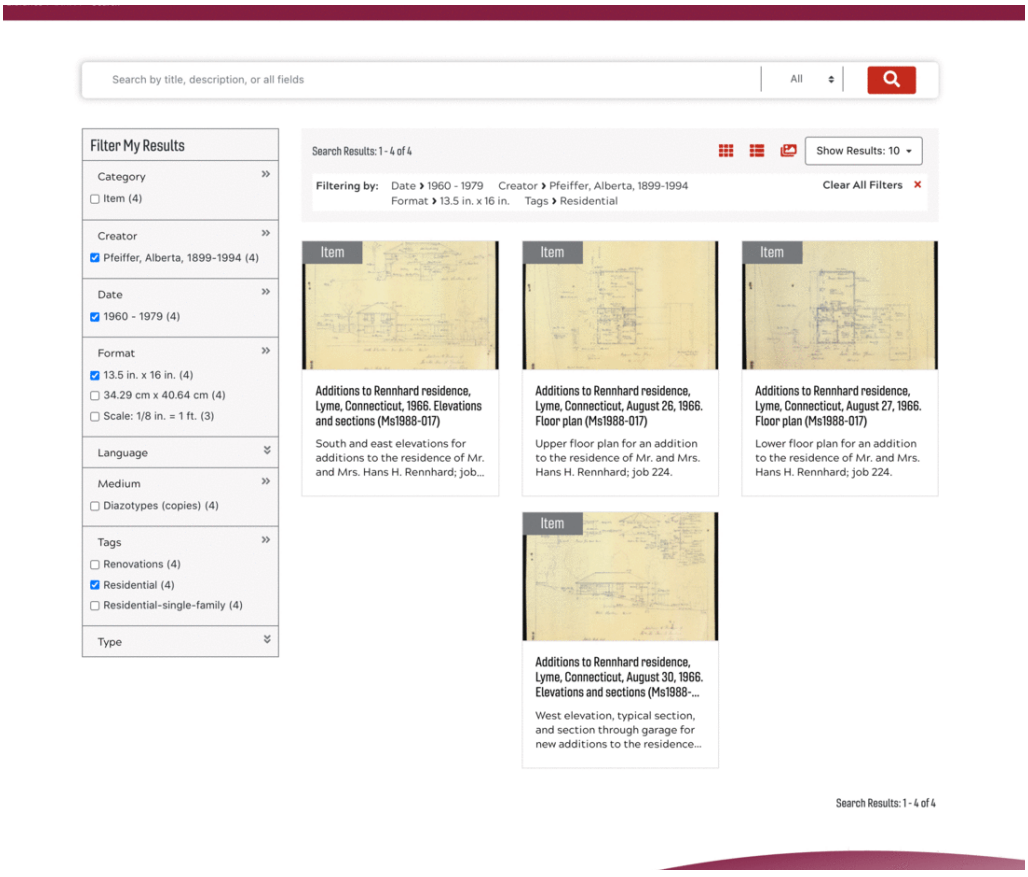

Figure 2: VTDLP Search Page (3)

In particular, the filter results sidebar contains several categories that simply do not apply to the type of data stored on the platform. Fields such as Format, Language, and Creator do not apply to the student submissions that we will be storing and enabling search/filtering on. Rather, we need to integrate our custom filters/search tags and categories that pertain more specifically to the students' submissions. We have outlined these searching tags/filters in Figure 3.

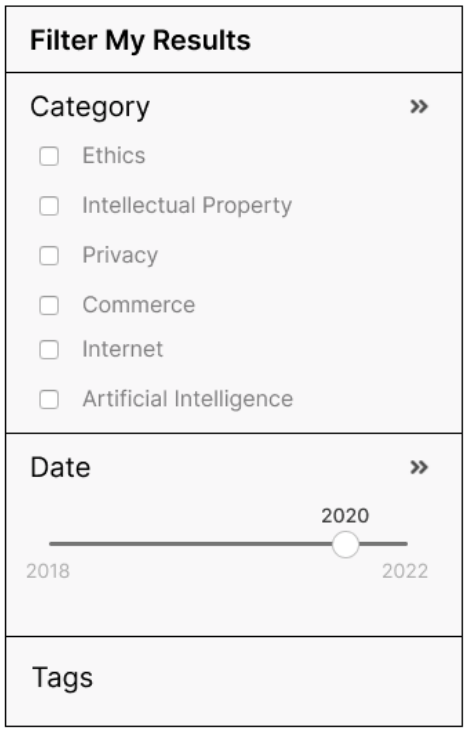

Figure 3: Digital Libraries' custom filters

Primarily, students will want to filter the presentations based on the category of the topic. These categories remain primarily unchanged from semester to semester; although, as was discussed between our team and our client, he retains the ability to add / edit / remove these categories. The next filtering metric will be based on the date of submission. This class has been ongoing for nearly a decade and will continue to

amass an ever-growing collection of student submissions. Thus it will be important for future students to be able to view submissions that are from older or more recent semesters. This feature also motivates some of the backend design on how student submissions will be processed, labeled, and stored in the collection database.

4.2 Wireframes

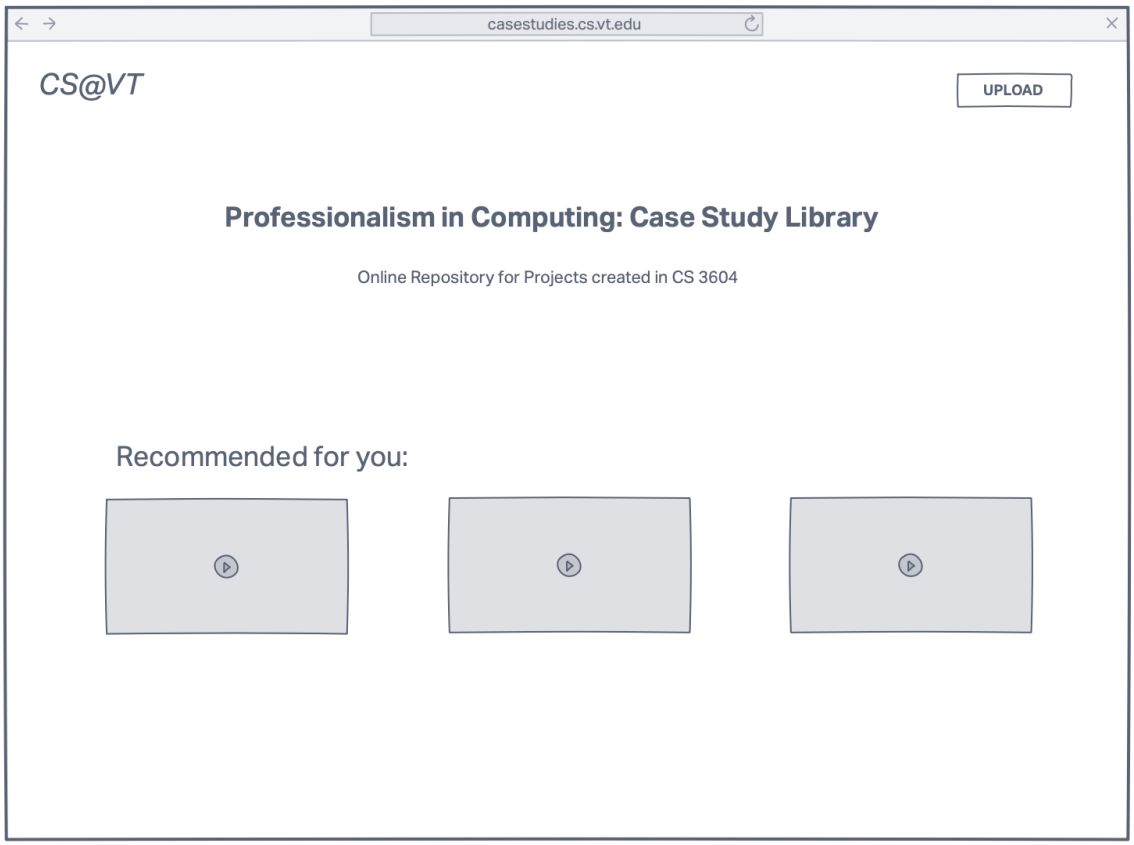

Figure 4: Wireframe of homepage

Figure 4 is a rough wireframe of how the web-app homepage will look upon completion. Notably, there is a section for recommended content that was requested by our client. This feature will enable greater exploration of the student project repository

by highlighting a small number of projects and incentivizing users to explore other projects.

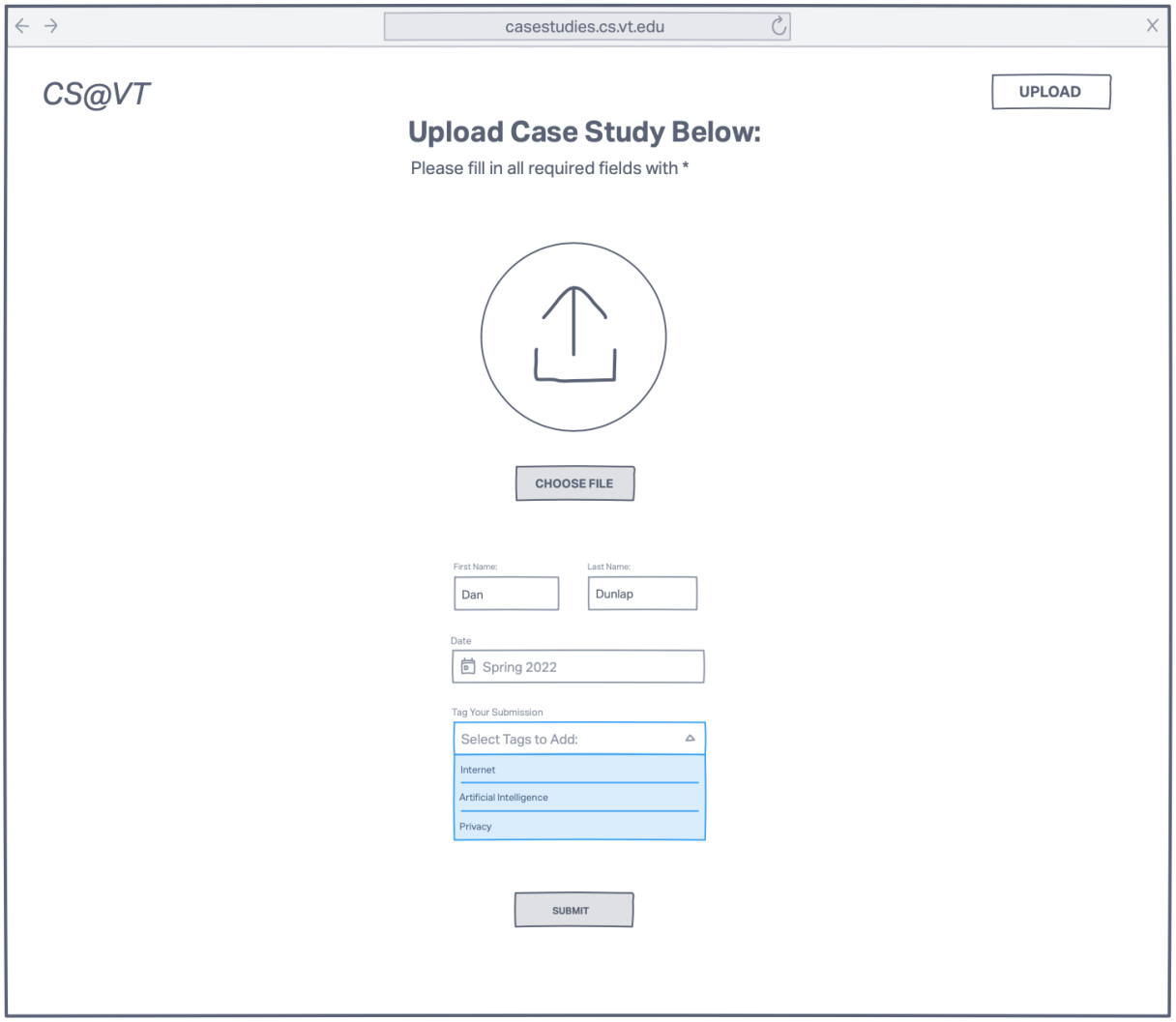

Figure 5: Wireframe of the upload page

Figure 5 shows a mockup of what the user experience for uploading new content to the website should consist of. This process is designed to be as streamlined as possible, minimizing confusion, and enabling fast uploads for all students. The students

will enter their identifying information (First and Last Name) as well as the semester of their current class section so that the presentation can be easily searched for. There is also a drop-down menu that is dynamically populated with the most up-to-date list of tags for the CS3604 class. Users will be able to categorize their submissions, making it easy for future students to filter presentations based on the category.

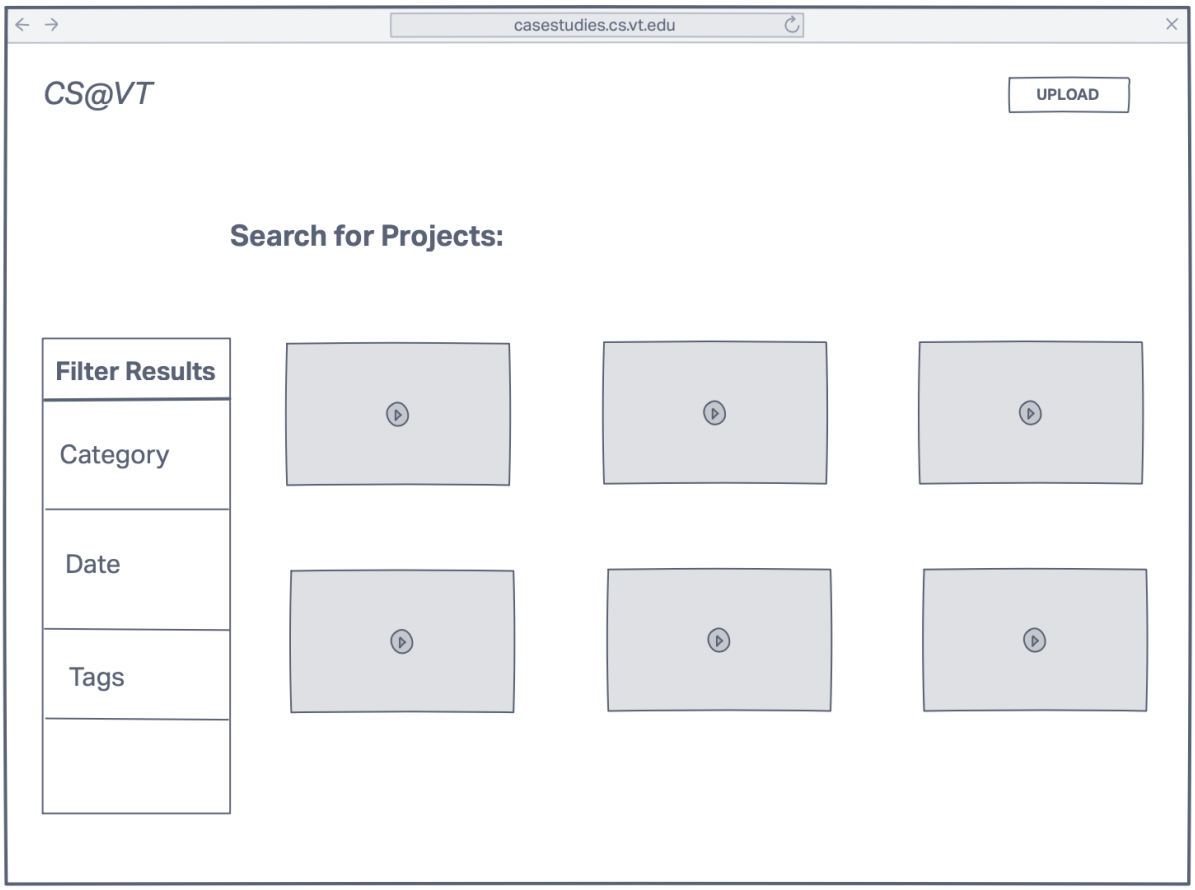

Figure 6: Wireframe of search page

Figure 6 illustrates the project search page which consists of a grid-like interface showing thumbnail previews of each project. These thumbnails will correspond to the category to which each project submission belongs, making it easy for students to at-a-glance

understand the general topic of each submission as they browse through the entire collection. On the left-hand side is the filter results pane within which users can categorically filter the submissions to only see the ones relevant to their desired search query.

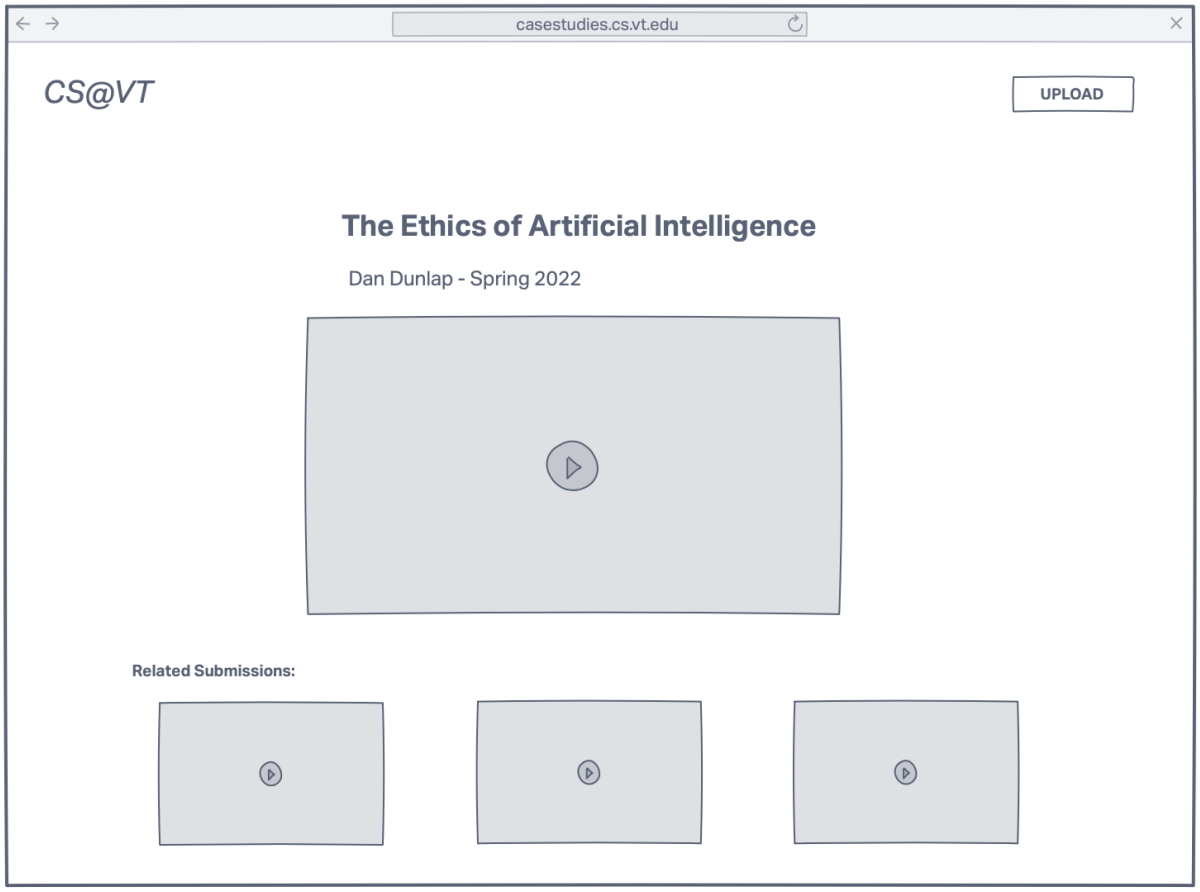

Figure 7: Wireframe of viewing page

Figure 7 shows a wireframe mockup for the individual submission page. This page will be rendered for each project submission, serving as the one-stop-shop for users to view and download prior student submissions. Also shown on this page is a related submission section that recommends users view other submissions of similar interest.

5. Implementation

This project was created for CS3604: Professionalism in Computing with Dr. Dunlap. In this course, students are required to create a “case study” that can take the form of a PowerPoint, MP4, or other files. In the past, these case studies have been uploaded to various sites such as Google Drive, YouTube, Dropbox, etc. Therefore, our goal of making a platform where students can view CS3604 resources would only make sense as a web application. Based on the template provided by the Virginia Tech Digital Libraries Platform, our team was able to create the CS3604 Case Study Library. Several of the main features of this platform include the ability to view previous semesters' case studies and search by various tags.

The first feature implemented was the ability to view previous semesters’ case studies. When the project was first announced, our team was given gigabytes of data from each semester which our team was able to parse and upload to the website. A large part of having a centralized digital library is that all previous case studies can be viewed. Each of these case studies includes the student’s name, their semester, and a brief title/description that explains their project. These case studies are also searchable. The main search bar allows users to search specific fields, presentations, and even creators.

Following the development of the web application itself, there remain several bureaucratic steps that need to be taken to ensure the ease of access and continuous availability of this application to VT students for years to come. First and foremost, we

need to work alongside the VT CS department to acquire a cs.vt.edu subdomain that will serve as the domain for this project. Beyond just acquiring the domain, we also need to work with the department head to acquire the necessary AWS licensing agreements that will allow for the collections to be hosted in perpetuity (4). While storage costs for this size of content are incredibly affordable, there does exist the potential for bandwidth charges to grow depending on the volume of traffic that is directed at this website.

6. Evaluation

To test the digital library, user testing was conducted between team members. As prior CS3604 students, it was easy to pinpoint the shortcomings of the course, and think about ways to improve. As such, our testing consisted of 2 main workflows that users may encounter. The first task asked users to search for a specific presentation using the search bar. The second task asked users to view a presentation. These workflows were timed and allowed us to understand the perspective of a CS3604 student. From the team's feedback, certain UI elements, such as the filter view and home page, were updated to improve efficiency.

We were able to meet a majority of our project deliverables and goals despite the challenges with AWS. The website was deployed and we were able to get a custom domain name at https://casestudies.cs.vt.edu/ (note: the Permissions tab still shows 'Access Denied'). The website's front-end was edited to have better aesthetics and improve the user's experience. We were also able to assemble a repository of the existing data and upload the presentations to the website for viewing. The files are categorized into different collections based on the semester they were submitted. For searchability, we were able to support searching based on specific keywords as well as the presenter's name.

7. User's Manual

To start using this digital library, a user can access it by going to the link https://casestudies.cs.vt.edu/, then the user is shown the homepage. The homepage, as shown in Figure 8, contains a search bar, a short description of what the website is, featured items, and a general highlight of what the library contains. The search capabilities on the homepage are limited to search by title or keyword.

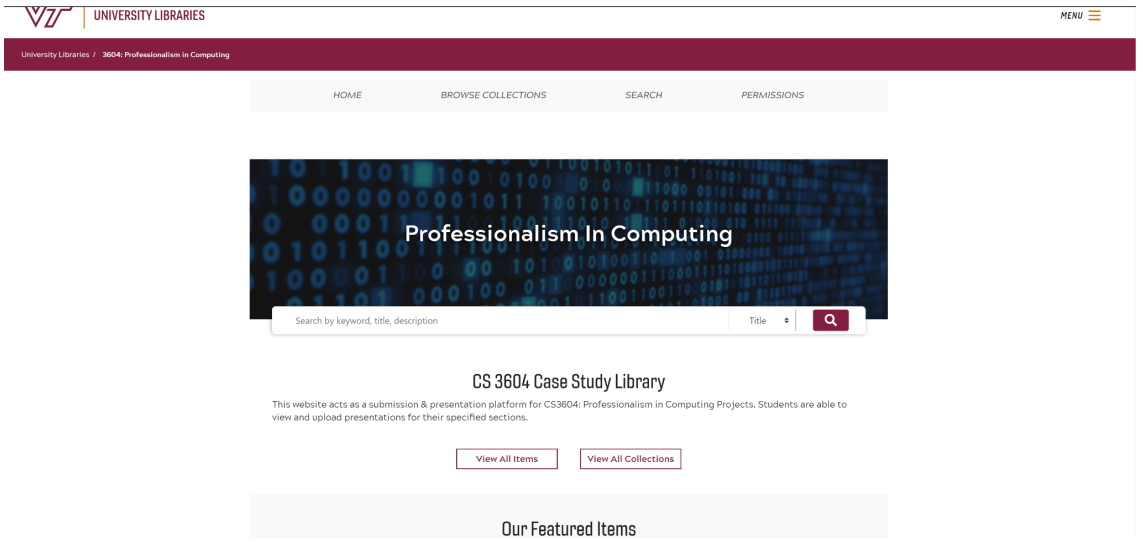

Figure 8: Homepage

The user also has the option to browse all collections that have been uploaded to the library. The user can get this part of the website by clicking "BROWSE COLLECTIONS" on the homepage. As shown in Figure 9, the user can then choose to view all the available collections or filter them by subjects from the dropdown list. There is also an option for the user to either sort these collections by alphabetical order, reverse alphabetical order, or by date.

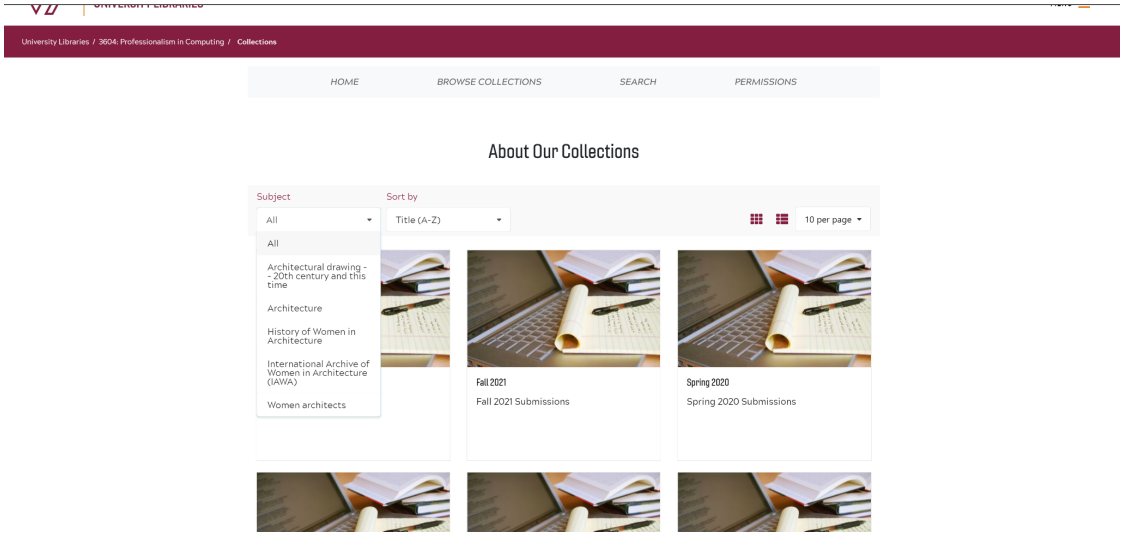

Figure 9: Browse Collections page

The user can perform detailed searches of the available collections by navigating to the search page of the website shown in Figure 10. This can be done by clicking the “SEARCH” button found on the homepage. The user is also redirected to this page when they perform a search on the search bar found on the homepage. On the search page, there are filters the user can apply to further specify their search. These tags include category and date.

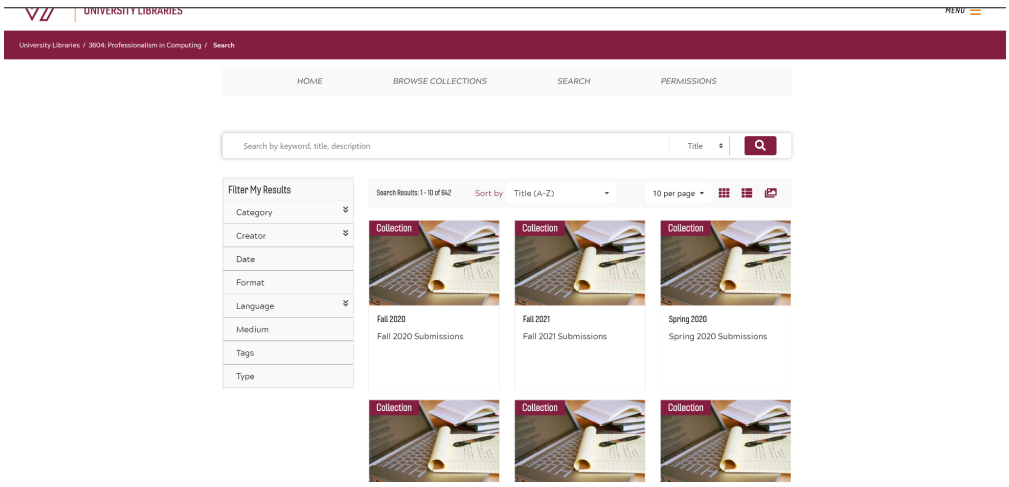

Figure 10: Seach page

8. Developer's Manual

This project utilizes the VTDLP application created by the Virginia Tech library (5). This application gets deployed onto Amazon Web Services (AWS) and takes advantage of serverless products that AWS provides. This means that no management of virtual machines, operating systems, upgrades, etc., is required by the developer.

To update the application's code, all that is needed is to commit to the "cs3604" branch in our GitHub repository, and Amazon Web Services will see the new commit, start building the website using the latest code changes, and then deploy it automatically (6).

If someone wanted to deploy our application or their version of VTDLP, then first they would need to fork either our repository or the normal VTDLP repository. Next, the forked repo should be cloned onto the developer's computer. Next, the developer should ensure that NodeJS is installed on their computer (7). Next, the developer should install the AWS amplify cli. This can be done by running

$ npm install -g @aws-amplify/cli

After this completes, the dependencies for the project should be downloaded. This can be done by changing into the dlp directory and running

$ npm install

Once this completes, the Amplify project should be initialized by running

$ amplify init

The last command that needs to be run within the terminal is

$ amplify push

This will start the first initial deployment onto AWS. This step will deploy the required resources in the cloud, and the status can be viewed from the AWS CloudFormation console.

Next, the developer should log into the AWS Amplify console. They can select to configure their repository to the deployment and add the front-end hosting. During this, environment variables should be configured. The default environment variables values that should be used are

REACT_APP_REP_TYPE=Default
USER_DISABLE_TESTS=true

Next, the redirect settings should be modified such that all paths that are not assets get mapped to index.html since the application is developed as a single page

application. This can be done by going to the Redirects and Rewrites tab and creating a new redirect with the following attributes:

Source Address: </^[^.]+$|\.(?!(css|gif|ico|jpg|js|png|txt|svg|woff|ttf)$)([^.]+$)/>

Target Address: /index.html

Type: 200 (Rewrite)

After configuring the rewrite, the site configuration should be uploaded. This can be done by navigating to the AWS DynamoDB console and selecting the Tables tab. Next, several tables should be displayed, including one that starts with “Site-”. If you click this table, then “Explore table items”, and then “Create Item”, the developer can upload the example site configuration (8).

Once the site is up and running, a custom domain name can be added so that users can access the website through an easier domain rather than the one provided by Amplify. For example, our original Amplify URL is https://cs3604.dpxazclvfrksv.amplifyapp.com/, and we have the custom domain of https://casestudies.cs.vt.edu mapped to our website instead. To add a custom domain, the developer should navigate to the AWS Amplify console and select the application they wish to add a custom domain for. Then, select the Domain Management tab, and an add Domain button should be seen. The developer can then enter the domain that they wish to use, and instructions should appear. These instructions will include information to add DNS records such as a verification CNAME record, as well as a

normal CNAME record from the custom domain wanted to the Amplify URL provided. Once the DNS records have been propagated, and Amplify gets the SSL certificate using the verification record, users should be able to securely access the site using the custom domain and HTTPS.

The developer can also modify some of the site's content without needing to make a deployment. This can be done through the site's admin page, which can be accessed at http://site-url/siteAdmin, such as https://cs3604.dpxazclvfrksv.amplifyapp.com/siteAdmin, as seen in Figure 11. An admin user can then log in and edit the website's content.

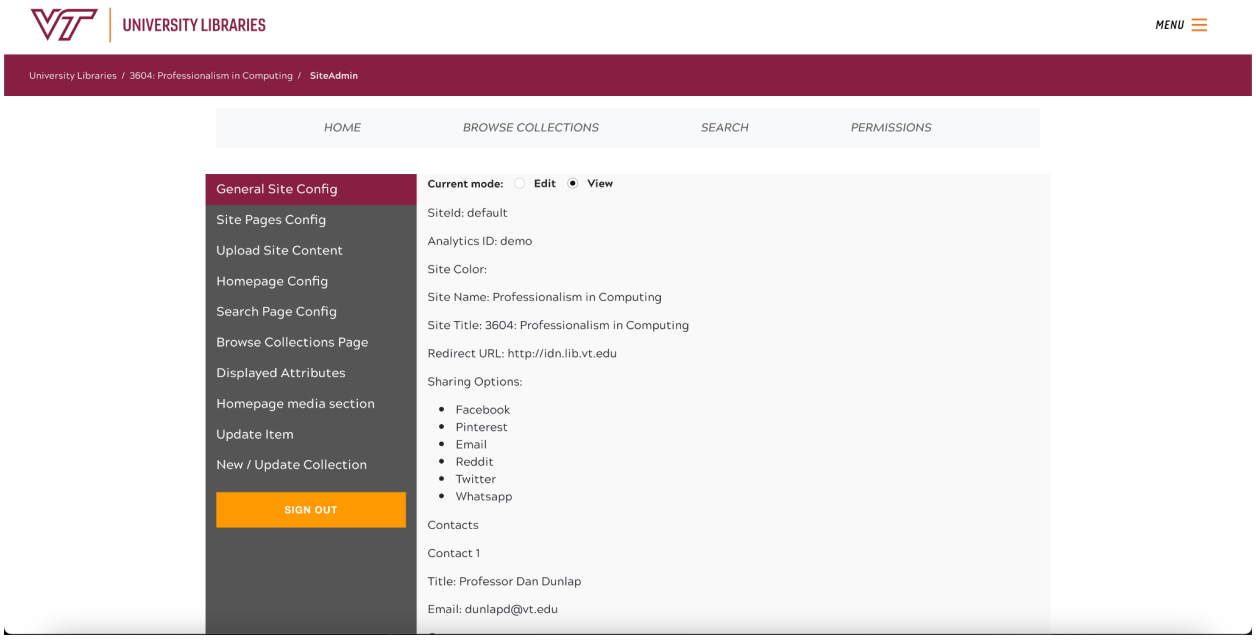

Figure 11: VTDLP administrative site

From this page, the admin can edit the site's configuration, upload new pages, edit the homepage, customize other pages, change the filters available when searching collections, upload new items, edit existing items, and create new collections.

On the general site config tab, the admin user can modify the site title, name, color, and contacts, as seen in Figure 12. Additionally, it can be specified which sharing options are presented to visitors of the website.

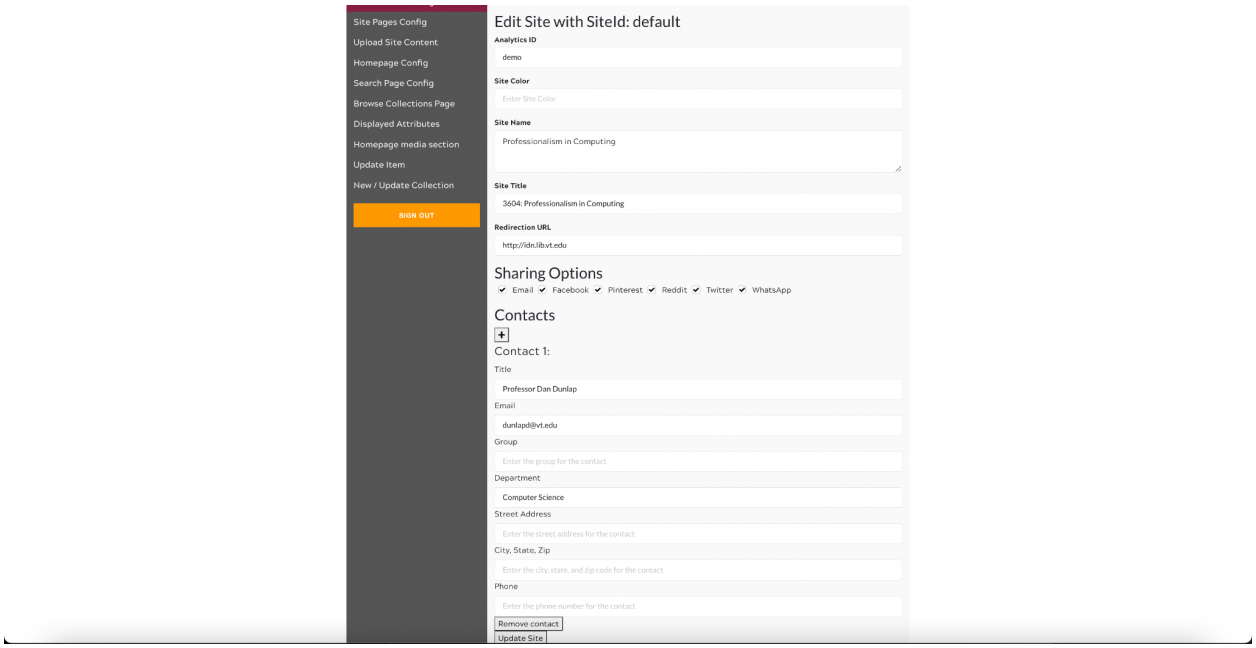

Figure 12: VTDLP general site configuration

Next is the site pages config tab, where the admin user can create new pages, and customize the IDs, permissions, components, assets, the URL that the page is accessed from, link text, and data URL for each page (the actual HTML content for the page). New pages can be added, or existing pages can be deleted, as seen in Figure 13.

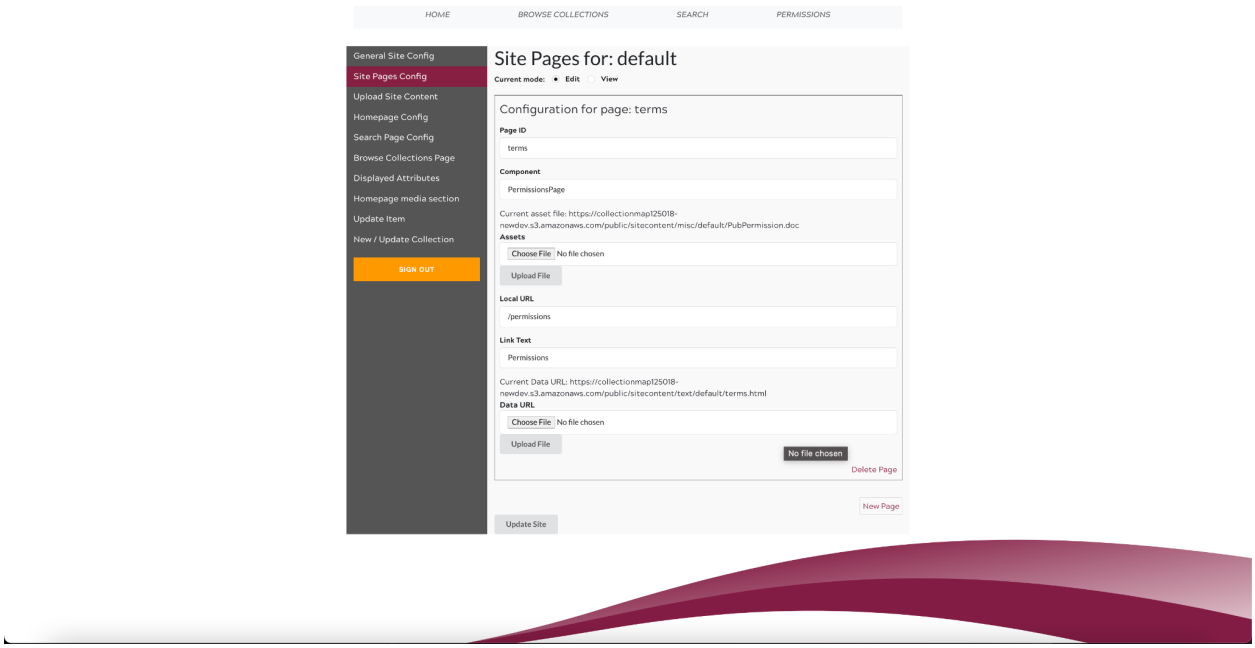

Figure 13: VTDLP administrative site pages

The next tab is the upload site content tab. This is where the admin can upload new images or HTML pages, as seen in Figure 14.

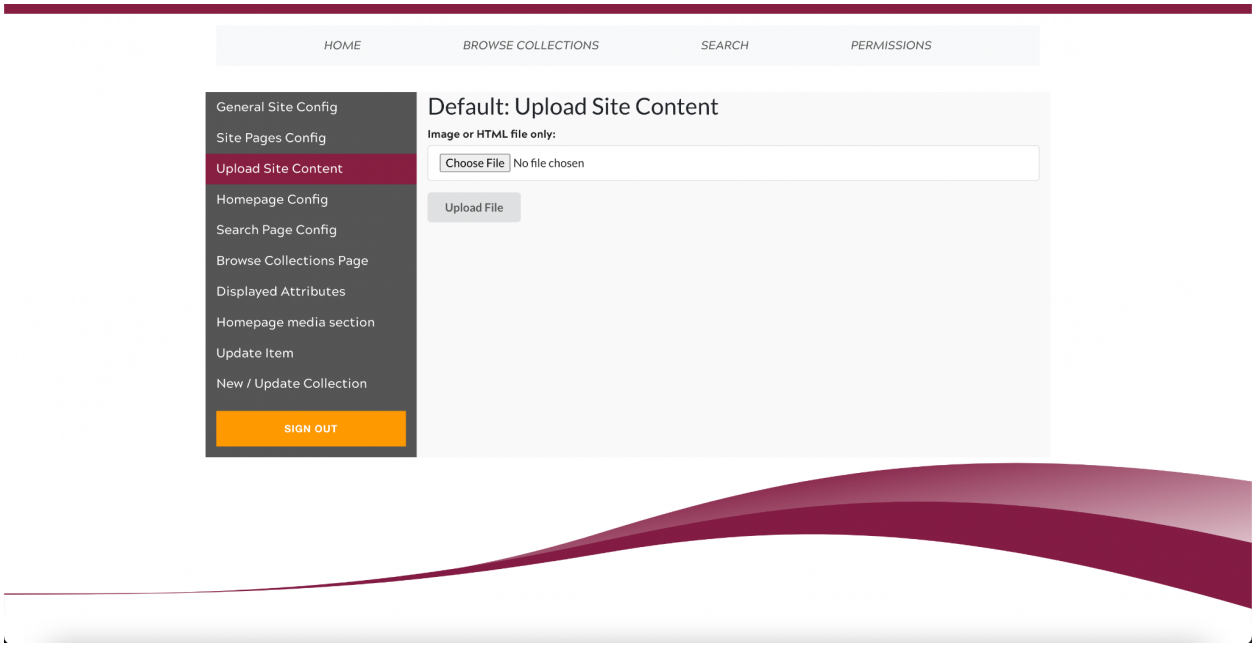

Figure 14: VTDLP administrative upload site

After the upload site content tab is the homepage config tab. This is where the admin can customize the wording, links, style, images, sponsors, featured items, and more. This is the go-to tab for when you want to change any of the content that is on the landing page of the website, as seen in Figure 15.

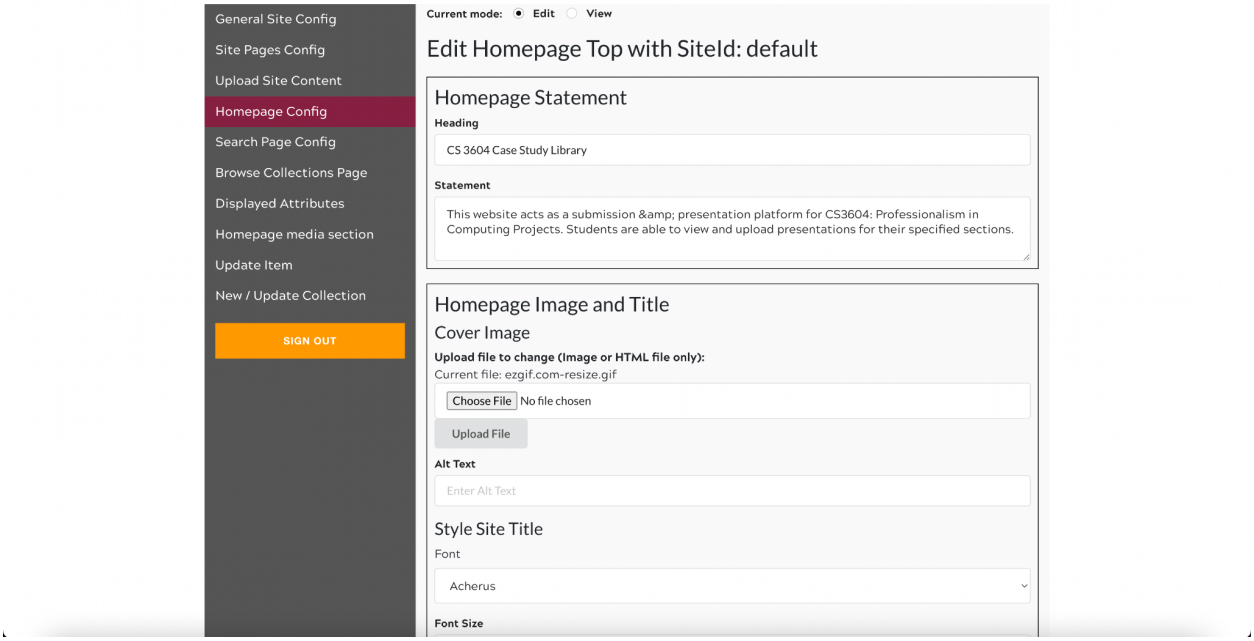

Figure 15: VTDLP administrative edit homepage site

The next tab is the search page config tab. This is where it is determined which fields can be used to search for items and the available options for searching, as seen in Figure 16.

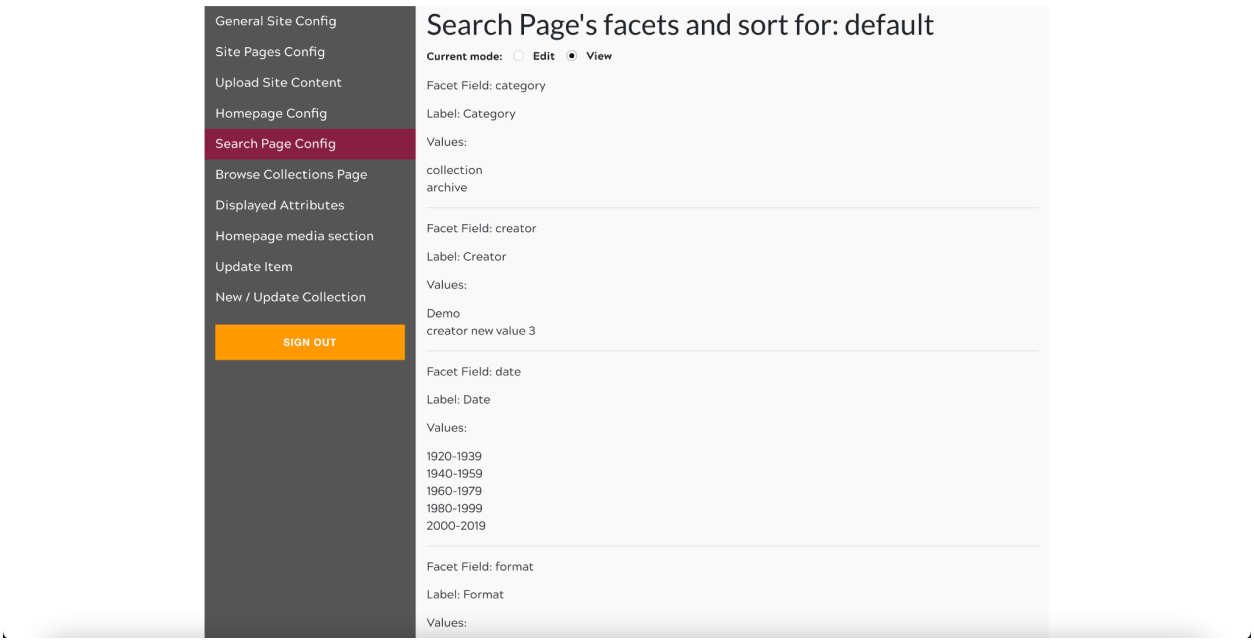

Figure 16: VTDLP administrative sorting page

The next tab is the browse collections page. This page is where the filters and sort parameters can be selected. This contains the options that change which filter categories and fields that can be used to search in categories, as seen in Figure 17.

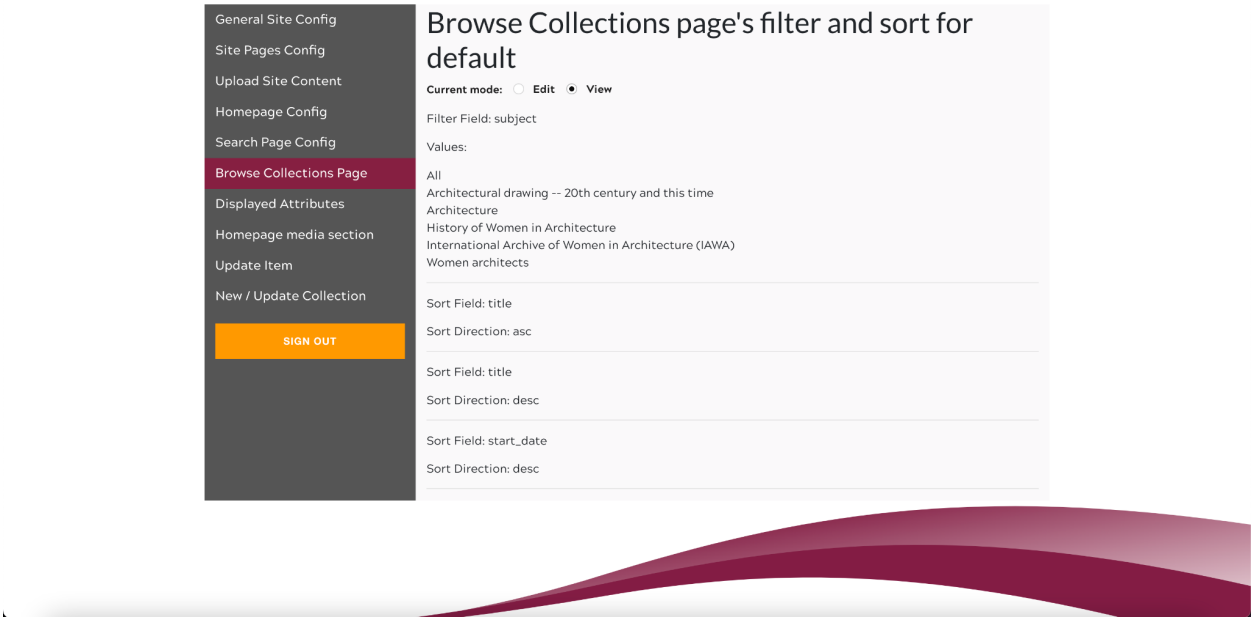

Figure 17: VTDLP administrative filtering site

The next tab is the displayed attributes tab, which is where the admin can select which attributes are displayed for all collections and items, and their display text for each attribute. This would be where the admin can add new attributes to each item, such as a semester, or topic, as seen in Figure 18.

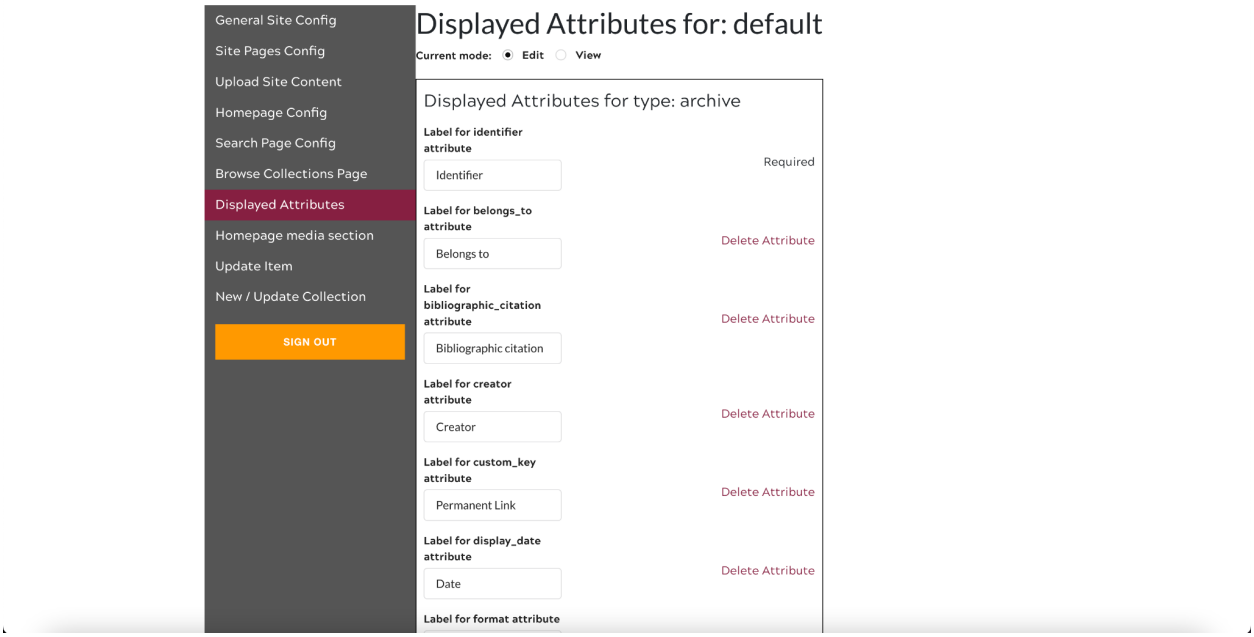

Figure 18: VTDLP administrative archive site

Next, we have the homepage media section, which is where the admin can customize the media section of the homepage. This is where a video and other text content can be displayed on the homepage for visitors to view. An example of what this section looks like is in Figure 20; Figure 19 illustrates the admin page of the homepage media section.

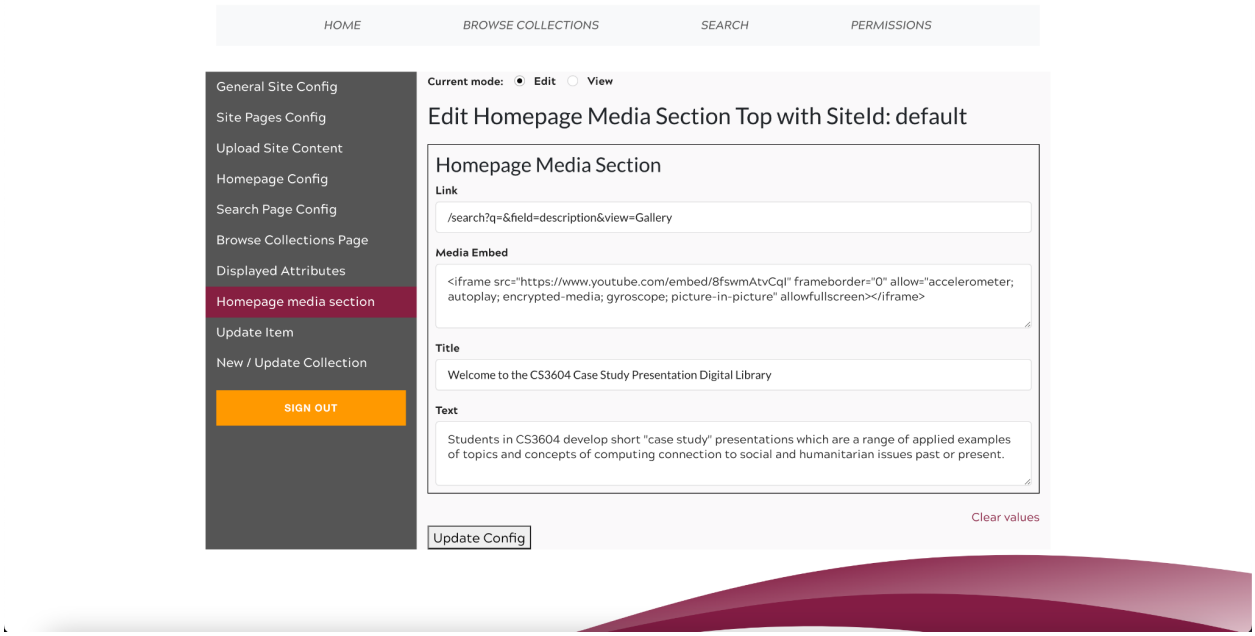

Figure 19: VTDLP administrative media site

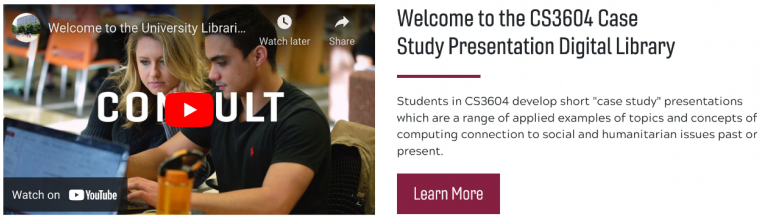

Figure 20: VTDLP display of media site

The second to last tab is the update item tab. This is where the admin can update the data of an existing item that has already been uploaded to the site. This could be used to add tags or correct existing data on an upload from a student or past semester upload.

The last tab is the new / updated collection tab. This is where new collections can be created, to display new items. The admin can decide how they want to organize their data, with a collection per semester, a collection per topic, or just a single collection with

all the data in it. They can create new collections, or adjust the title or other information of existing collections, as shown in Figure 21.

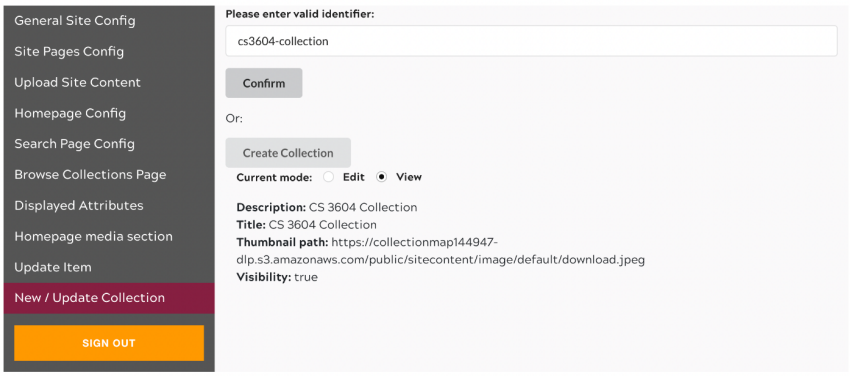

Figure 21: VTDLP administrative collections site

8.2 Professor Automatic Upload Tool

In addition to the development of the integrated VTDLP platform web app, we worked on building our own automated archival batch upload script. The purpose of this tool is to aid professors in automatically uploading large amounts of student data to the site all at once.

8.2.1 Configuration

Before the automatic upload tool can be used, some setup must be performed on the client's device.

MacOS

curl "https://awscli.amazonaws.com/AWSCLIV2.pkg" -o "AWSCLIV2.pkg"

sudo installer -pkg AWSCLIV2.pkg -target /

aws configure

At this point, the professor must enter the provided AWS credentials

Pip install urllib.parse boto3

Windows

Download the AWS CLI MSI Installer for Windows (64-bit)

https://awscli.amazonaws.com/AWSCLIV2.msi

OR

Alternatively, you can run the msiesxec command to run the MSI installer.

msiexec.exe /i https://awscli.amazonaws.com/AWSCLIV2.msi

Aws configure

At this point, the professor must enter the provided AWS credentials

Pip install urllib.parse boto3

8.2.2 Usage

Using the script is designed to be a simple and intuitive process. Place the script in the same folder as your desired upload directory.

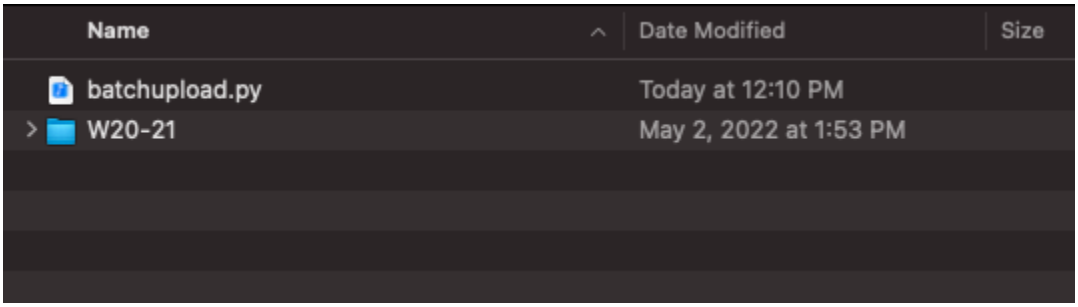

Figure 22: Example Folder Structure

Open a terminal and navigate to this file directory by using the cd command:

Cd Documents/Path/To/Desired/Folder/Destination

Execute the script using the format below:

Python batchupload.py PathToUploadFolder/

Follow the on-screen prompts as shown below to specify the necessary fields/parameters for upload and then await program completion.

```
Kyles-Macbook:test kylepapili$ cd Documents/Path/To/Desired/Folder/Destination
Kyles-Macbook:test kylepapili$ ls
W20-21          batchupload.py
Kyles-Macbook:test kylepapili$ python batchupload.py W20-21/
Enter a Name For This New Collection:
Winter 2020-2021
Enter a Description For This New Collection:
Winter 2020-2021 Submissions
Uploading 9 out of 10
Complete
```

Figure 23: Example Script Execution

8.2.3 Design

The current version of the automatic upload script was designed for simplicity and ease of large quantity uploads. This script automatically generates titles for each submission based on the PID and filename of the student file. Furthermore, the script does not enable the professor to manually input categorical or descriptive data for each individual submission as this would be a long and tedious manual process. The metadata included with each submission includes the file name, file size, submission date, submitter PID, and an automatically generated submission title.

8.3.4 Performance

The speed at which files can be uploaded depends greatly on the user's internet connection and total available bandwidth. AWS supports file uploads in the S3 storage tier of up to 100Gbps so most users will be limited by their own local internet connection.

The script provides a progressive readout of upload progress so that the user has an idea of how much time is left; although, upload times vary from submission to submission as a .mp4 video submission will be a much larger file than a .pdf submission for example.

9. Lessons Learned

9.1 Timeline

Table 1: Timeline of the Project

Month	Work Completed
February	Received AWS account from Dr. Chen. Worked with him to get the required permissions for using the account.
March	Deployed the barebones site. Catalog the data in the zip files. Received permissions to work on the front-end of the website. Started working on the python script to upload the files to the website.
April	Continue to work on the front-end overhaul of the website. Curate the archived data so that it is in the correct format for the Python script to mass upload them. Deploy the website on the custom domain.
May	Move the website onto the new AWS account. Use the Python script to upload the files to the website. Put the finishing touches on the front-end of the website.

One of the most useful guidelines for ensuring consistent progress throughout this project has been a willingness and determination to adhere to the timeline; see above. At the start of this project, we set out to design our timeline in a way such that the time-based goals are very broad which left us flexibility in the order in which we went about pursuing more nuanced milestones. Had we tried to be specific with the order in which we approached smaller milestones, we likely would have found ourselves

struggling to maintain that order of priorities. Often, it is not until you are fully immersed in solving a problem that you best understand the most effective way to go about it.

9.2 Problems

We were surprised to learn that the majority of problems we encountered during this project were not about technical complications but rather had to do with bureaucratic negotiations and deliberations.

Most notably, we were significantly delayed in starting technical development on this application as we were awaiting the CS department allocation of an AWS cluster. We also had trouble having the proper permissions to correctly deploy certain aspects of the website. To make any edits on the front-end, we needed the permissions to access the admin page (shown in Figures 11-18) and make changes. User-testing with the current students in the CS3604 class was stalled as it was challenging to get the proper permissions to deploy a collection on the website where we could upload the existing data. We wanted to ensure that we had the framework for submitting data to the website set up and that we allowed for searching by the desired fields. This delay meant there was little time left for there to be user testing in the CS3604 class.

Additionally, there is 6.5GB of data with around 500+ submissions within a single year, from the semesters between Fall 2020 to Fall 2021. Uploading a semester's worth of submissions would take around 3-5 minutes to upload. All of this archived submission data received from Dr. Dunlap was given in a variety of formats described in Table 2.

Approximate Submission Format Distribution	
MPEG-4 Video	38.75%
PowerPoint (pptx)	36.25%
Google Slide Links	11.25%
PDF (pdf)	8.75%
QuickTime Movie	5%

Table 2: Approximate Submission Format Distribution

9.3 Solutions

In close collaboration with Dr. Chen, we were eventually able to secure a 6-month temporary AWS cluster on which we could complete the development and deployment of the digital library platform. We once again collaborated with Dr. Chen to receive the proper permissions. At each step along the way, we had to receive permission from Chen to move on to the next section of the project. Dr. Fox and Dr. Dunlap also obtained an AWS account to redeploy the website. The website has been approved to receive funding from the CS department to maintain the AWS account and the website.

9.5 Future Work

The majority of future work includes user testing. While we were successful in building the infrastructure to allow the digital library to be functional, there is still more work to be done to improve the users' experience. Due to complications with AWS, we were not able to let the current students of CS3604 use the website. In the future, user testing could involve allowing the students to have access to the website and being able to have the correct permissions and interface to upload their presentation documents to the website directly. It's our goal to make this platform the primary resource for future semesters, which will only be possible with extensive testing/iteration.

One specific feature that remains to be implemented is the ability to search by tags. In CS3604 students can make a case study about a few topics including Intellectual Property, Privacy, eCommerce, Information and Communication Technologies (ICT / Internet), and Machine Learning/AI/Algorithms. Each of these topics needs to be given a tag, which will allow users to search projects much more efficiently.

For the user upload feature, our proposed UI is as follows: To upload to this library, a user will navigate to the upload page by clicking the "UPLOAD" button also found on the homepage. The user will then be redirected to a page where they upload a file of their choosing, add their first and last names, and pick the appropriate tag that describes their topic.

An additional future feature to be implemented is a greater degree of manual control in the automatic batch upload script. As it stands now, the professor has no ability to manually specify a category, description, or title for each submission. This can be resolved by prompting the user to input this data manually for each file submission or by accepting a .csv file as input that would contain the relevant information already laid out.

On the website page, the permissions tab currently just shows “Access Denied.” In the future, this can be updated to allow site admins to access the data or to allow students currently enrolled in CS3604 to submit files to the website.

The cost for running this website on AWS depends on how much it is used. For DynamoDB, which is the database used in the application, it costs $0.25 per GB per month to store. For Amplify, which is the website hosting, costs $0.15 per GB transferred, and $0.023 per GB stored per month. Also, the application uses Amazon Elastic Search, with a t2.medium.elasticsearch sized VM, which costs $0.073 per hour. It also uses AWS S3 to store any uploaded files, such as images, presentations, etc, which costs $0.023 per GB per month. As of 5/7/2022 after uploading the 6.5 GB of submissions that we have from the past, our current month forecast within the AWS Billing Dashboard is $5.70. Future work analyzing cost trends once the website is in use by real students during the semester would be better at predicting the costs going into the future for the website.

10. Acknowledgments

We would like to give a special thanks to Professor Daniel Dunlap for allowing us to help him in developing the CS3604 Case Study Library. We would also like to give a big thank you to Dr. Yinlin Chen for his help on the deployment of the VTDLP Platform. Finally, we would like to thank Dr. Edward Fox for facilitating this project and providing us with invaluable feedback throughout the semester. Further, Professor Calvin Ribbens, Department Head of Computer Science, authorized support for our work to be extended using departmental funds for an AWS account.

Dr. Daniel Dunlap

Email: dunlapd@vt.edu

Dr. Edward Fox

Email: fox@vt.edu

Dr. Yinlin Chen

Email: ylchen@vt.edu

11. References

(1) Scott McCrickard, Deborah Tatar. "CS3604: Professionalism in Computing." Department of Computer Science, Virginia Tech, 2022, http://cs.vt.edu/Undergraduate/courses/CS3604.html. Accessed 13 Apr. 2022.

(2) Digital Library Development Team, "Virginia Tech Digital Libraries Platform," 2022, University Libraries, Virginia Tech, Blacksburg, VA 24061 USA, https://about.digital.lib.vt.edu/. Accessed on 4/21/2022.

(3) IAWA. “IAWA | Search.” International Archive of Women in Architecture, Virginia Tech University Libraries, https://iawa.lib.vt.edu/. Accessed on 5/4/2022.

(4) Rangel, Derek. “DynamoDB: Everything You Need to Know About Amazon Web Service's NoSQL Database.” Amazon, 2015, https://aws.amazon.com/dynamodb/. Accessed on 4/21/2022.

(5) Chen, Yinlin, et al. “VTUL/DLP-Access: DLP Access Website: A Multi-Tenancy Serverless Web Application.” GitHub, 2019, http://github.com/VTUL/dlp-access. Accessed on 4/21/2022.

(6) Chen, Yinlin, et al. “VTUL/DLP-Access: DLP Access Website: A Multi-Tenancy Serverless Web Application.” GitHub, 2022, http://github.com/BrianWieder/dlp-access. Accessed on 4/21/2022.

(7) Node.js. Node.js, OpenJS Foundation, 2021, http://nodejs.org/en/. Accessed on 4/21/2022.

(8) Chen, Yinlin. “DLP-Access/Site.json at CS3604 · Brianwieder/DLP-Access.” GitHub, 2021,

http://github.com/BrianWieder/dlp-access/blob/cs3604/examples/jsons/site.json. Accessed on 4/21/2022.