CS4624
Multimedia, Hypertext, and Information Access

2022-12-15

CS3604 Case Study Library II

Virginia Tech, Blacksburg, VA 24061

Instructor:
Dr. Edward A. Fox

Client:
Dr. Daniel Dunlap

Team:
Trevor Bagbey, Matthew Betsill, Omar Elgeoushy, Daniel Setareh

Table of Contents

Table of Figures

Table of Tables

1.0 Abstract

Examples of applied knowledge are vital to any university student looking to develop a deep and abiding understanding of their major. Such examples pour the foundation of changing the world through novel, thought provoking innovations and advancements by offering insight into how an idea can turn into a reality. This Case Study Library was developed with the purpose of providing students and others a litany of various cases in which specific Computer Science topics were relevant in industry.

Case studies, in this context, are the multimedia presentations by students of Virginia Tech in CS3604: Professionalism in Computing that take place at the end of the semester. Students are instructed to pick an example from industry pertaining to the Internet, Artificial Intelligence, Intellectual Property, Commerce, or Privacy. Through thorough research and the learnings from the class itself, the students construct their presentations on their topic of choice.

This project is the second iteration of the Case Study Library. From the previous project group's work, the library held more than 500 case studies by students of Professionalism in Computing. It was evident that there were some major aspects that could be improved upon. These included titling the case studies by file name, no classification of case studies by course topic, and no thumbnail images. Along with this, a significant percentage of files were unable to be displayed due to file format issues. The burden of uploading case studies was on the professor, who needed to run a Python script to batch upload site items. This iteration of the project had a solid understanding of what needed to be done, including the addition of a student upload page, stylistic corrections, search parameter specification, and thumbnail images for files. Search filtering, student authentication, collections by course topic, and functionality to upload more than one case study file were also added. Changes to the site were made via a frontend administrative page as well as making additions and modifications to the code base.

With the Case Study Library having been improved, it now stands as an effective tool that current students of Professionalism in Computing can reference while they work on their own case studies.

2.0 Introduction

2.1 Client

CS3604: Professionalism in Computing is a required course for students majoring in Computer Science at Virginia Tech. The class focuses on the study of the many ethical, social, and professional concerns that are raised within the field of Computer Science [1]. The semester is divided into blocks covering broad Computer Science related topics, and students are expected to select their own case study to research and present from within that course topic. Primary topics covered in CS3604 include Artificial Intelligence, Privacy, Internet (ICT), Commerce, and Intellectual Property. Students deliver their case study as a five minute multimedia presentation at the end of the semester. The class is taught by Dr. Daniel Dunlap, the client of this project.

2.2 Problem

The Professionalism in Computing Case Studies are done every semester. In the past, many of these presentations have been repeated. Along with this, it was also found that the quality of the case study projects themselves could be improved. With those issues in mind, there needed to be a way for students to have examples of prior work to guide their decisions pertaining to the project. This would hopefully create more unique case studies of an average higher quality. Cases are also only shared within class sections. This means that some students are not exposed to beneficial topics that could pique their interest if a certain interesting case study was done by a student from a different class section.

2.2 General Approach

For this project, we were tasked with maintaining and improving the CS3604 Case Study Library built by the previous semester's CS3604 Case Study Library team [2]. The ultimate goal was making the site easier to navigate, such that CS3604 students could easily use the site to view previous case studies. One issue which made the site difficult to use was the categorization of case studies by the semester in which they were delivered. We decided categorizing instead by course topic made case studies much easier to navigate, given the large and ever-growing number of them on the site. Another issue raised was the lack of a simple method for the addition and upload of new case studies. The only method of upload from the previous semester was a Python [3] script for batch uploads which did not allow the uploader to submit case study specific metadata. Our solution to this was to implement an upload page allowing students to upload and

categorize the case studies themselves. Finally, our client for this project, Professor Dunlap, requested an audit of the finances involving the Case Study Library to see whether current hosting and service costs could be minimized.

The team initially focused on solving the maintenance related issues plaguing the CS3604 Case Study Library. This included checking for outdated dependencies, security issues, unnecessary expenses, outdated API keys, and other general improvements. Collections were instantiated for course topics, and an upload page was created for student case study submissions. Lastly, the pre-existing case studies in the library were removed and re-submitted to include the needed metadata of a name, course topic, title, and description.

Through the organization and collation of hundreds of prior semesters' case studies, this project allows anyone to view and browse CS3604 Case Studies from prior semesters. This is a great benefit to current and future students as well as interested alumni. Current and future students taking Professionalism in Computing can look towards the CS3604 Case Study Library for guidance towards the creation of their own case studies for their semester. By having access to this library, the quality and topic breadth of case studies can be improved.

Currently, the site is being hosted on Amazon Web Services (AWS) [4] and can be accessed at the following domain: https://casestudies.cs.vt.edu/ [5]. 500 dollars is allotted by the CS Department at Virginia Tech [6] to provide for hosting and domain costs related to the library.

3.0 Requirements

The nature of the student case studies in the Professionalism in Computing course makes paramount the accessibility of prior information to guide students through selecting a topic and creating their own case studies. Through meetings with Dr. Dunlap, some general improvements upon the prior iteration of the project were discovered to be potentially beneficial.

3.1 Accessibility

Overall, the site needs to showcase the work of prior students in a way that could be easily consumed by students and alumni alike. For students, it would need to be used to research previous projects that could expand their knowledge on certain topics, eventually helping them choose a case study of their own design. For alumni, the cases could provide an interesting point of view of current students on relevant topics.

For the service to be used efficiently, case study categorization needed to be implemented. This would greatly increase the students' access to relevant information by allowing them to search by topic without sifting through hundreds of presentations and their titles. Along with this, the frontend of the application needed to be updated to correct some general issues with rendering images and styling. Finally, case studies needed to have titles. The Python batch uploading script from the previous semester only titled the case studies by filename, which offered little help when searching.

3.2 Student Upload

Most importantly, students should have the capability of uploading their own case studies to the platform. This relieves the burden of batch uploading and individually categorizing the case studies from Dr. Dunlap and instead puts it into the hands of the students. Also, as previously mentioned, having individual uploads allows for more case study specific information to be input. This information includes course topic, title, and description, which play a large role in how students can navigate through the site. Similarly, given the recent change in the course that requires students to sum their case study presentation up in a paper, we have allowed up to two files to be submitted with a case study upload, which can either account for the student's slides and paper, or the student's MP4 video and slides. Given this change to the course curriculum, case studies uploaded to the library will now not only have slides that can guide future students, but also a paper or video to fill in the context of the slides so that a better understanding of the case study can be conveyed. Upon submitting two files in a submission instead of one, each of the two archives needed to have an associated file link that can be followed to the other archive, allowing users to get a better picture of the overall case study.

3.3 Maintenance

As this is a second iteration on a project, general maintenance on the site was also necessary. If the site ran into an issue, which happened in the development this semester, it would have to be remedied. Another one of our goals was to help eliminate the need for frequent maintenance, which is built into the design of serverless applications [7]. Having simple user interfaces, such as an upload page, to access the storage and database services connected with this application, also helps mitigate maintenance needs.

4.0 Design

The design for the site has a simple outline in the VTDLP Access Website [8], which makes adding new collections and archives trivial. However, since our solution for creating a new student upload section involves making additions to the actual codebase, the frontend design must follow the same design schema used in the original site. This can be done by utilizing the SCSS [9] files and React Components [10][11] already included in the codebase for the case study upload page. The frontend design for the modified collections to represent course topics will be inherited from the current codebase.

As for the backend design for allowing case study uploads, there are three important concepts that need to be addressed for the upload functionality to work correctly. The first concept is the functionality of storing the actual file such that it can be accessed following the upload. To solve this issue, our approach involves using Amazon S3 [12] to store the file in the S3 bucket associated with the Amplify App [13]. Similarly, there needs to be a bucket policy [14] added to the specified directory in the bucket so files in that directory can be accessed publicly. Amplify offers a simple API [15] for adding new files to its associated S3 bucket, which is what is used in our solution. The next component needed is the ability to add the case study's metadata to the Archive DynamoDB [16] table. This can be solved similarly by using the GraphQL API [17] offered by Amplify, in which 3 things need to be passed: the query, which comes from the createArchive query associated with the App, an object representing the archive and its associated metadata, and the authentication mode to be used for the upload. This third parameter leads to the third and final component needed for a successful case study upload, which is authentication. For our approach, we use Amazon Cognito User Pools [18][19], which involves checking the given user's account for a specific group to access the page. Similarly, the permissions associated with this group are needed to successfully make an upload.

5.0 Implementation

The Case Study Library implementation relies heavily on the Virginia Tech Digital Libraries Platform, or VTDLP [20]. VTDLP is an open-source project that uses the AWS infrastructure with the goal of offering a full stack solution of a library database. The previous iteration of the project decided that it would be a good choice, as it is already specialized to exhibit research work by Virginia Tech students. The frontend of VTDLP is DLP Access [8], which is the codebase and AWS architecture that displays the library content itself. To make changes to the website, there are two methodologies. The first is through the site admin page [22], in which a user with the proper credentials can make simple alterations to the archives, collections, and pages of the site. The second methodology is to change the codebase itself, which gives more flexibility and creative freedom. While the previous team's changes were only made from the site admin page, we determined that in order to implement the changes we wanted, we would need to primarily focus on making additions to the actual code base.

Given that this is the second iteration of the Case Study Library, it is important to distinguish between the two implementations. The first group established the site itself, forking the DLP Access GitHub [8]. With that they created a novel S3 bucket [12] to store the case studies themselves, into which their Python script for batch uploads deposited the case studies. Along with this, they uploaded limited metadata for those case studies into DynamoDB [16]. To state clearly, new to the site this semester are the case study and collection thumbnails, the student upload page, collections based upon CS3604 topics, and search filtering by such topics. Along with those aspects, the case studies also now have titles, creators, and parent collections; display the upload date; and can have associated case studies as an associated link. PowerPoint files are now not allowed to be uploaded as the site does not support them, and videos must be in MP4 format. The S3 file storage as well as the DynamoDB instance were changed to suit the new needs of the site as well. Overall, we are proud to say that this implementation of the project is very usable and looks aesthetically pleasing, indicating a marked improvement over the previous semester.

5.1 Frontend

The frontend of the site works off the DLP Access open-source platform [8]. This is the frontend used for all VTDLP [20] applications. The architecture is shown in Figure 1. A user interacts with the frontend via a React Web Application [14] hosted on AWS Amplify [15].

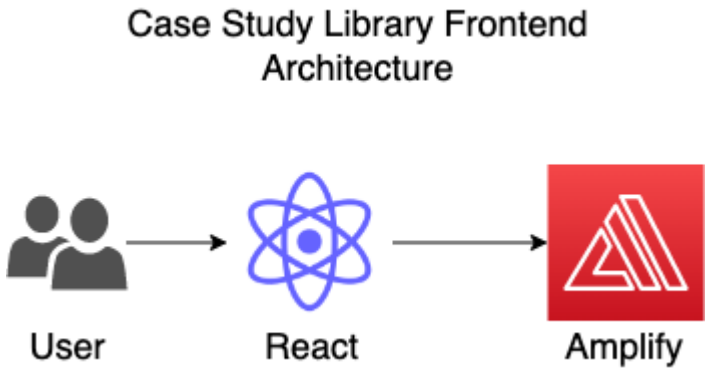

Figure 1: DLP Access frontend AWS architecture

To implement the improvements to the frontend of the site, our group decided to work off of some of the pre-existing React [11] code. The groundwork of this code came from the site admin page, which provides a multitude of functionality, including an interface for the site administrator to upload case studies. As the students themselves would need their own page for uploading, a combination of updating the codebase along with editing the site itself through the site admin interface gave us a good starting point.

The new student upload page was created via a new JavaScript [23] file for the page itself, along with a React component [10] that contained the functionality for the page. The Ant Design [24] Library was also added to the project to facilitate the frontend design. Simple validation is performed in the submission form to help prevent the delivery of erroneous submissions to the backend.

To provide security for the student upload, there needed to be authentication preventing access to the upload page by anyone not in Professionalism in Computing that semester. It was decided to use AWS Cognito [18] user authentication functionality similar to that of the site administrator authentication. This allows access to the upload page for specific accounts. Effectively, there is one account that will serve as a password for accessing the upload page. Students are given this account's credentials by the instructor, granting students the ability to upload their case study. Potential improvements on this design are

further discussed in the Future Work section of this report. This authentication grants the upload permission to the S3 bucket [12], as well as the GraphQL mutation [25] for the Archive DynamoDB [16] table.

5.2 Backend

The case study files themselves are stored in an S3 bucket [12] and the metadata for the case studies are stored in DynamoDB [16]. The metadata includes all necessary information regarding the case study including the course category it belongs to as well as the creator, upload date, description, and potentially associated links.

Once a case study has been uploaded to the S3 bucket, a lambda function [26] is triggered to generate a thumbnail image. This thumbnail is used on the frontend for display purposes. This is an improvement upon the previous iteration which used the same placeholder image for all displayed case studies. To allow the lambda function write functionality to the S3 bucket folder, a new bucket policy [14] statement was created specifying the Amazon Resource Name [27] of the lambda function as principle with write functionality.

A full representation of the storage and backend of our modified implementation of DLP Access [8] is shown in Figure 2.

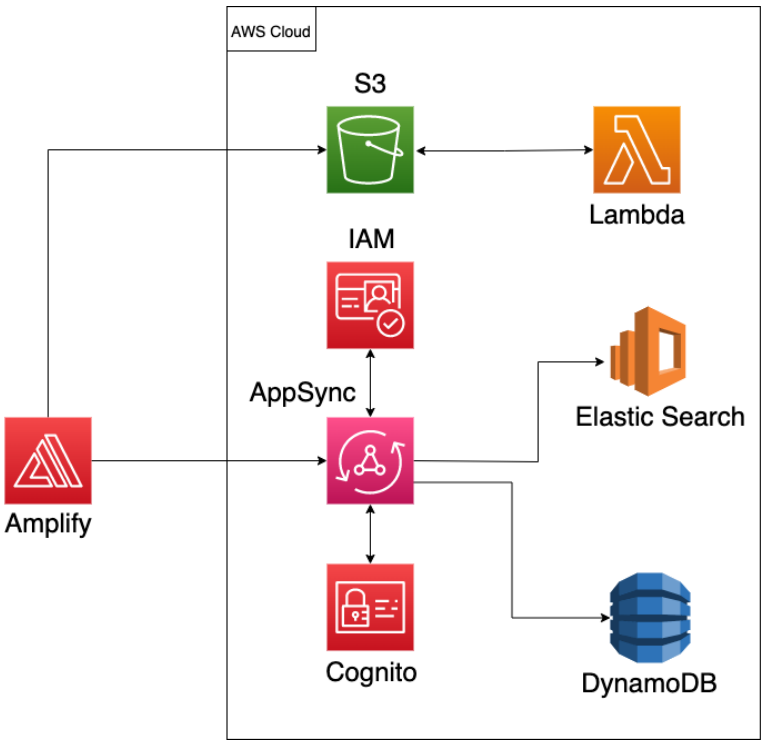

Figure 2: DLP Access backend AWS architecture

Case study searching is performed with AWS OpenSearch Service [28] via an ElasticSearch [29] instance. Case study and collection metadata from DynamoDB are

indexed in order to provide fast retrieval on the frontend. The access to DynamoDB write functionality is controlled via Cognito [18].

6.0 Evaluation

Every change performed upon the site was assessed and tested to ensure that regressions or errors were never introduced. In addition, our tests and evaluation performed upon the previous implementation of the Case Study Library revealed some errors, which we identified and marked as issues on our initial review.

In addition, extra thought was put into identifying the different kinds of users of the Case Study Library, such that the library fulfills their specific purpose for browsing it. We identified three major groups of users, being current students in CS3604, previous students of CS3604, and future students of CS3604. We then analyzed their specific needs from the library.

An example of this user-focused design process would be our option for an Anonymous Submission of case studies. As we expect students in CS3604 may want to keep their submissions anonymous, there is an opportunity to do so when utilizing the new submission feature. Not only do we account for this, but checking the Anonymous box reminds the student that they will need to remove their name from their project before submitting it to the library. By thinking about our designs from the perspective of the real-world user, we have been able to perform extensive evaluation and testing upon the Case Study Library. We would have ideally liked to have performed testing using current CS3604 students, but this was infeasible due to the scope and fluidity of the changes being made to the site.

The library currently holds case studies dating back to Spring of 2020, with an average of about 400 case studies being submitted per year. With the addition of our upload page and alterations made to existing case studies, all case studies on the site are now categorized by course topic and can be filtered as such when searching, or by visiting the “Browse Collections” page and going to the specified course topic collection. We anticipate that the Case Study Library will continue to be used for years to come, so long as the CS department continues to fund it.

7.0 User's Manual

7.1 Supported Functionality

The Case Study Library has the capacity to be used by anyone interested in real-life examples of how computing topics present themselves in industry. It is publicly available online via web browser and currently has over 800 case studies. The site has the ability to search via creator name, title, description, and collection. It also offers search filtering by collection.

For students of Professionalism in Computing, the site gives them the opportunity to upload their own case studies and metadata through the upload page. Students can also elect to upload an associated file to be displayed via link on their case study's view page. For referencing case studies, a permanent link is provided that gives access to the case study even if the site URL changes.

For Professors and TAs of Professionalism in Computing, the site gives them the ability to delete case studies or edit case studies that have previously been uploaded by students.

7.2 Searching

Users of the site may search through the various case studies using the search bar on the home page [4] or the browse collections tab, as shown in Figure 3. Searching may be done with respect to title, description, or all attributes of an archive's metadata.

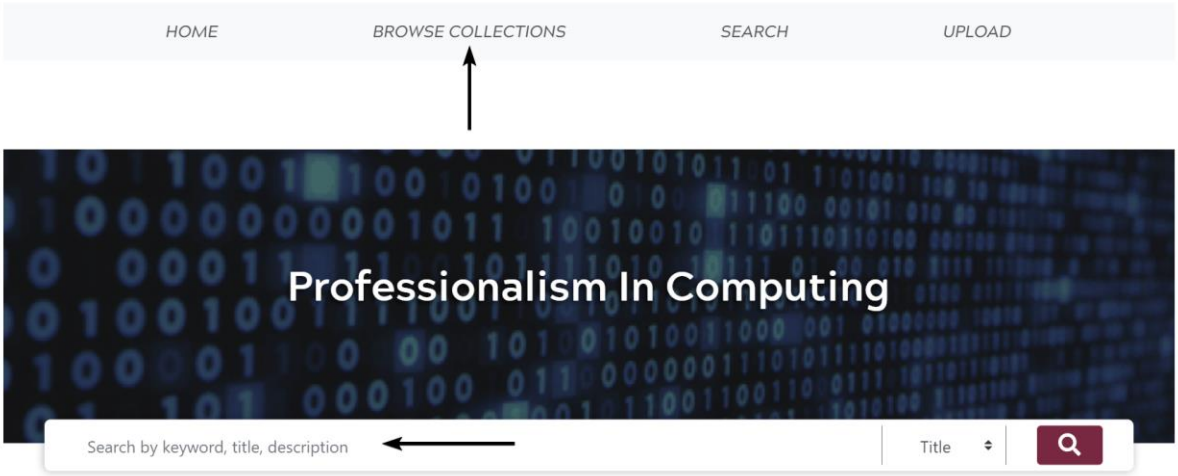

Figure 3: Methods of accessing search functionality

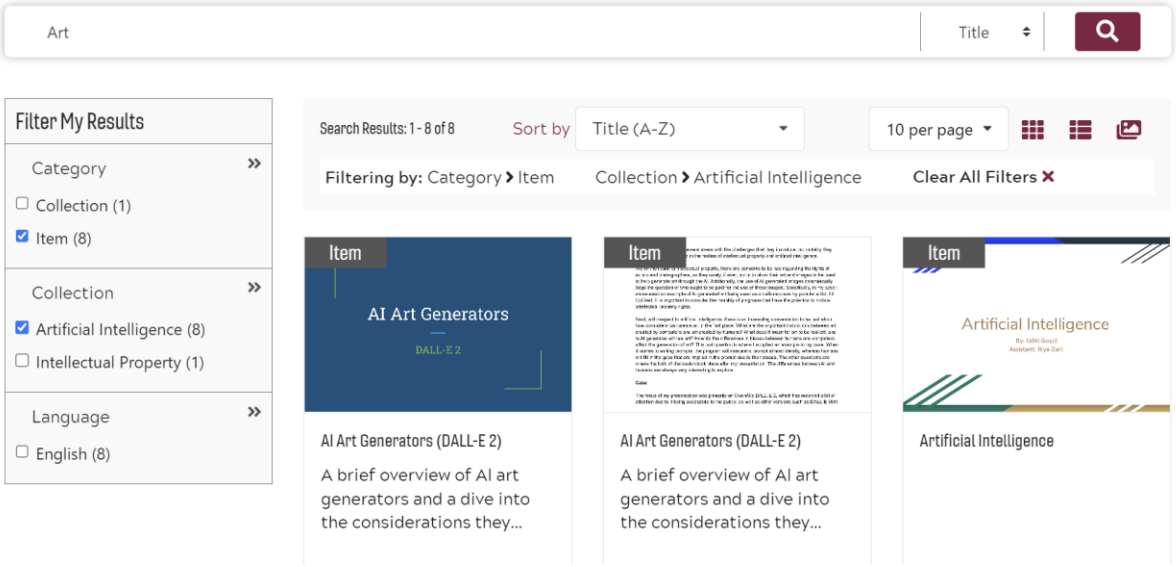

Figure 4: An example of a filtered search for 'Art'

Once the search has been completed, users may further their query by filtering the results by the course collection to which the case studies belong, as demonstrated in Figure 4.

7.3 Student Upload

To upload a case study, students should follow these steps.

7.3.1 Obtain Login Credentials

The username and password should be provided by Dr. Dunlap once case studies have been completed and are ready for public view.

7.3.2 Navigate to Upload Page

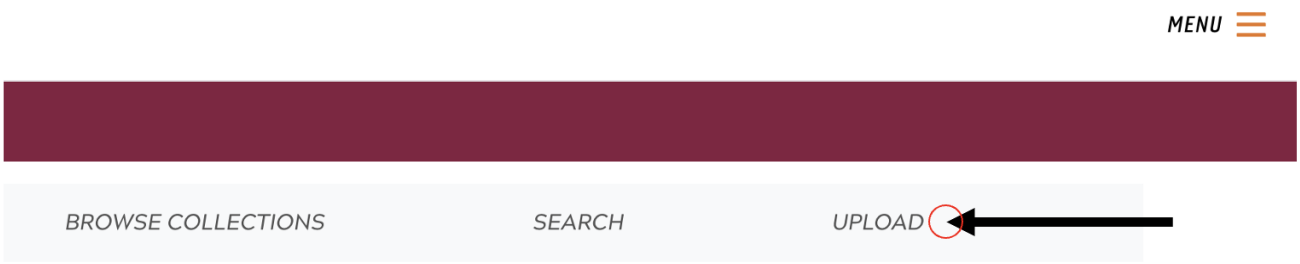

Figure 5: Student upload button

On full size screens, the student upload button will appear just below the header as shown in Figure 5. On smaller screens, the header buttons appear in the side menu, as seen in Figure 6.

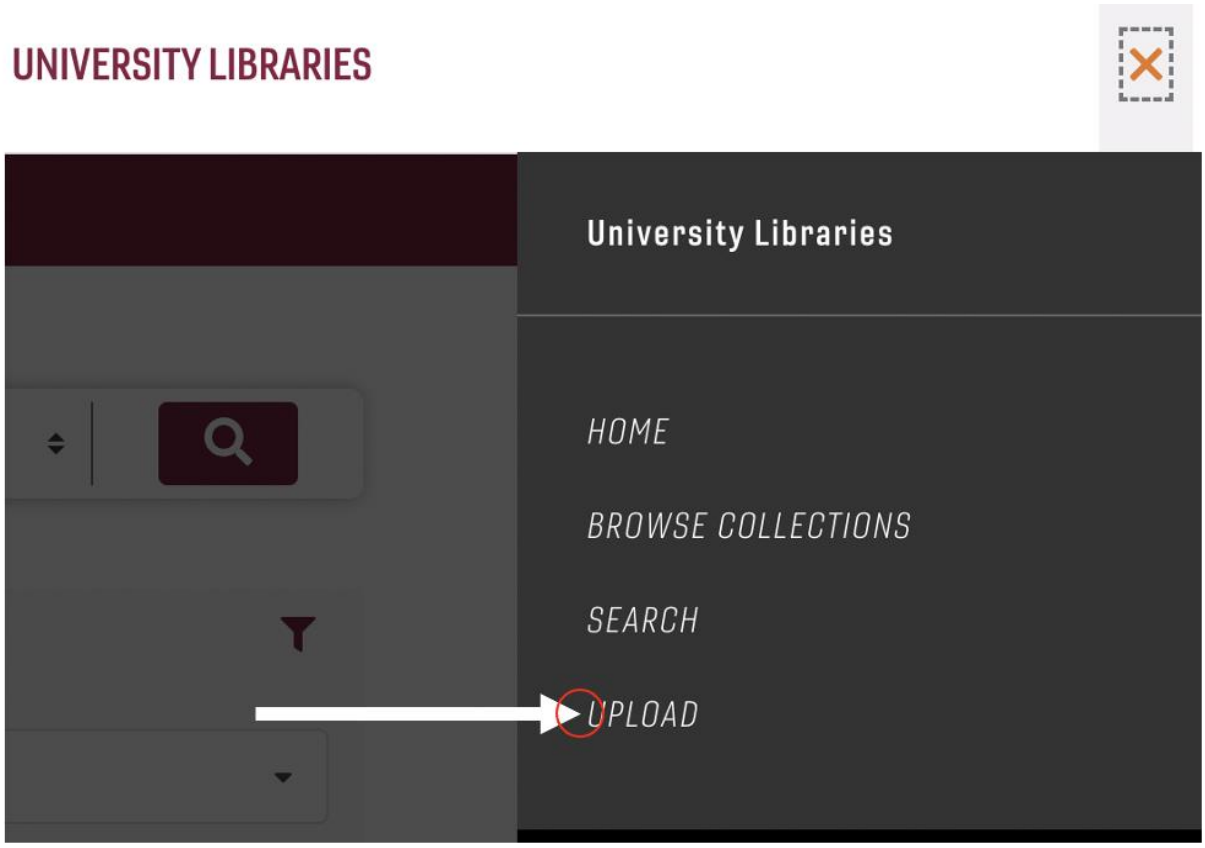

Figure 6: Student upload button (side menu)

7.3.3 Sign in Using Credentials

Once the upload button has been selected, a login card will appear that prompts the user to enter login credentials. The login appears as shown in Figure 7. This is where the student will enter the credentials provided by Dr. Dunlap to give access to the upload page itself.

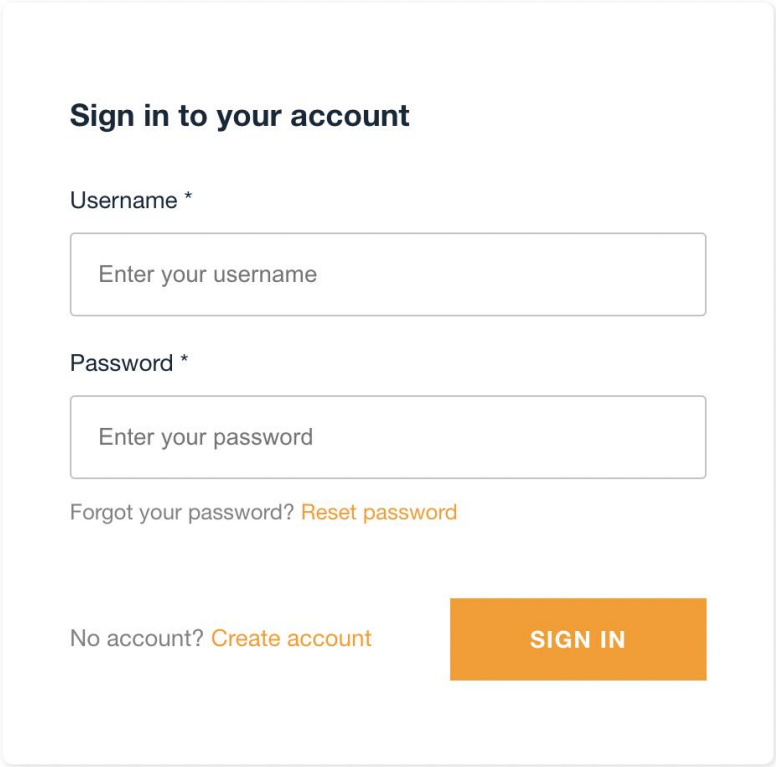

Figure 7: Student upload login

7.3.4 Fill out form with necessary information

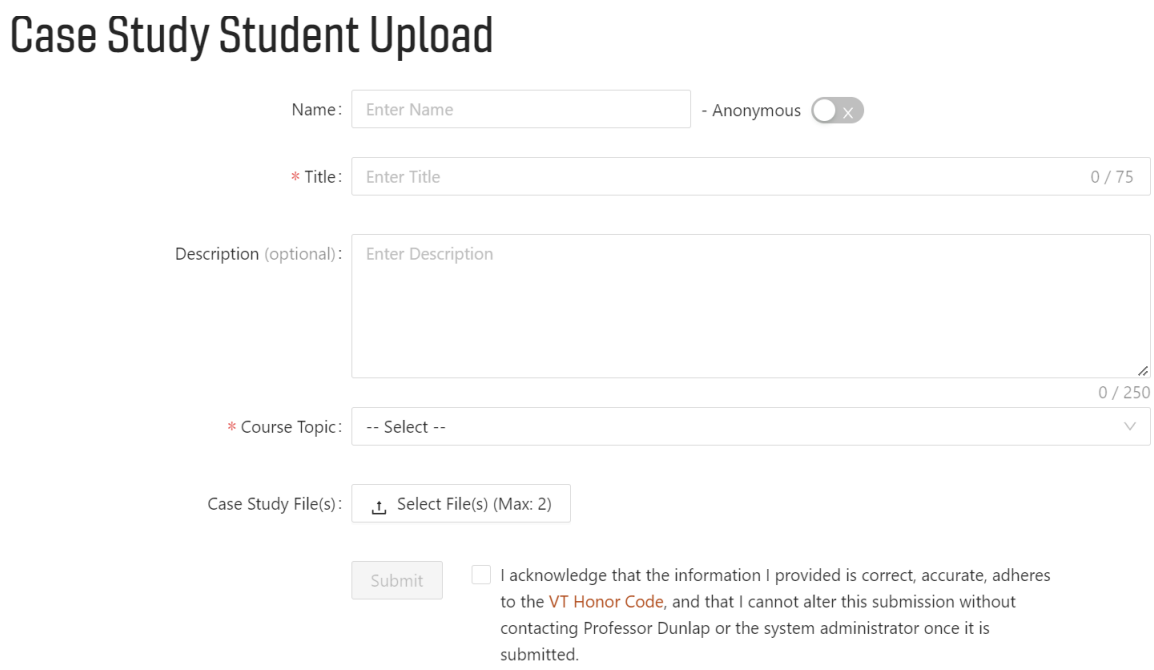

Figure 8: Student upload fields

Once login credentials have been verified, students are presented with the student upload page itself. First, the student can choose to upload the case study anonymously or with their name. The title of the case study, the course topic, and the case study file itself are all required for submission, as shown by Figure 8. The submission does not go through unless these conditions are met. A description may be added to help further the likelihood that the case study will show up in a search query.

To ensure students will not falsify information or plagiarize others' work, before submitting, a student must agree to the VT Honor Code [30]. Along with the Honor Code, students also agree to understand that their upload will not be editable once submitted. Presumably in rare instances, a site administrator may be able to help if mistakes were made. By having students check the box, we try to limit the likelihood of that happening.

7.4 Case Study Editing and Deletion

For a case study to be edited or deleted, a TA or Professor with administrator credentials can use the following steps.

7.4.1 Navigate to the Site Admin Page

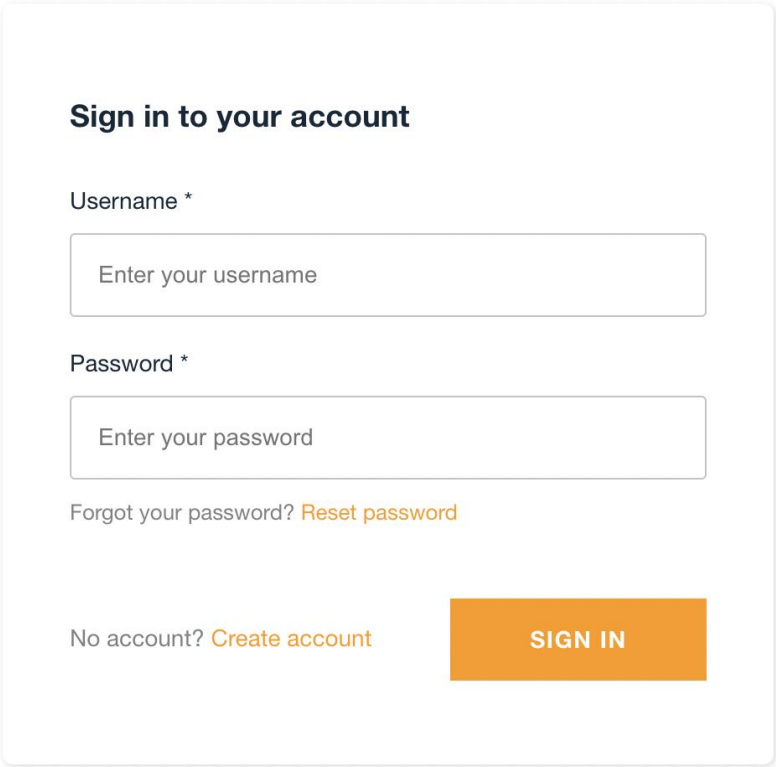

Figure 9: Site Admin Login

Navigate to the site admin page at https://casestudies.cs.vt.edu/siteadmin. Once there, enter administrator credentials into the fields shown in Figure 9.

7.4.2 Select “New/Update Item”

Next, an administrator should select the “New/Update Item” tab on the menu, as shown in Figure 10.

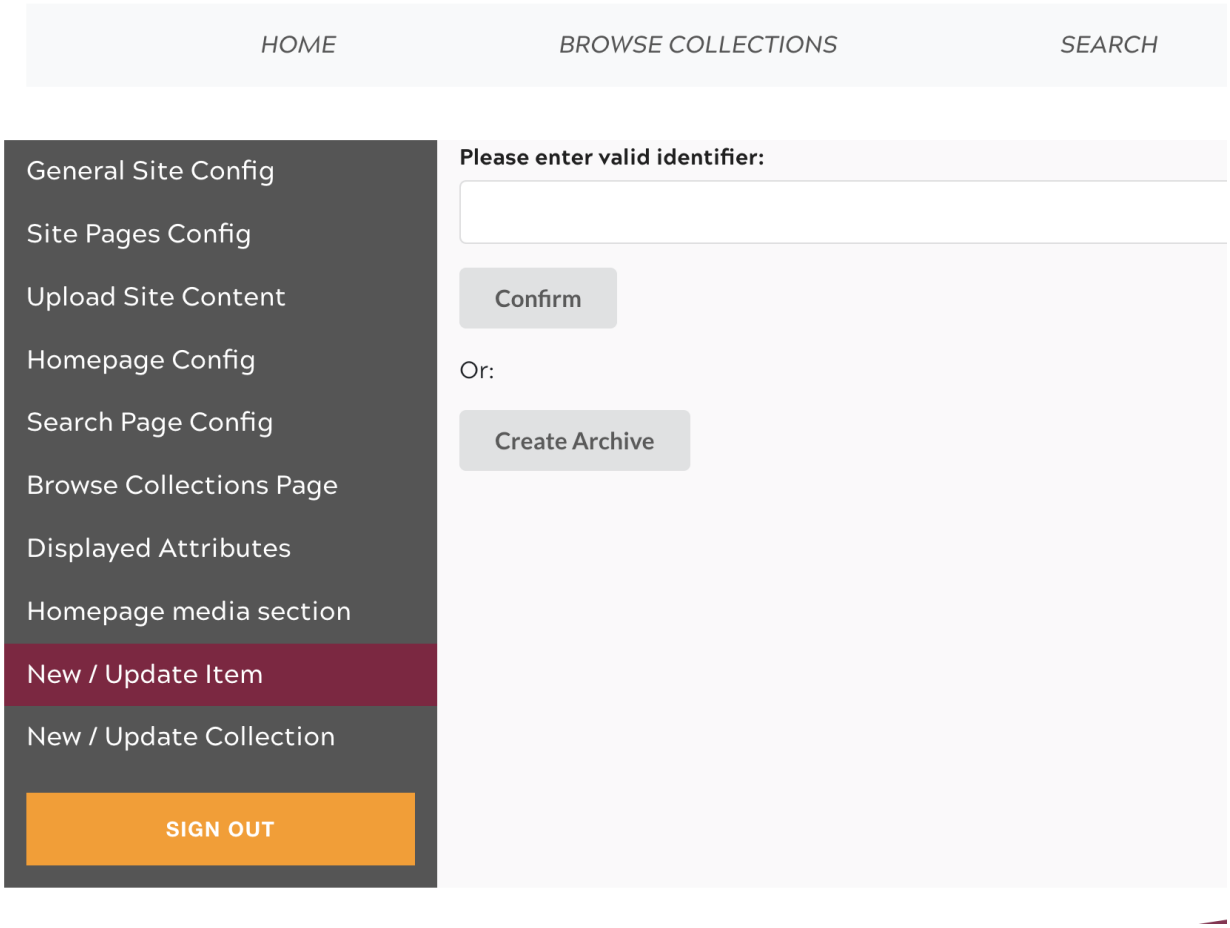

Figure 10: Site Administrator New/Update Item

7.4.3 Enter Case Study ID

After selecting New/Update Item, enter the case study key of the case study which you would like to delete or edit. This is located on the page of a case study, as shown in Figure 11 under the label “ID”. Once input in the field labeled “Please enter valid identifier:”, select the “Confirm” button.

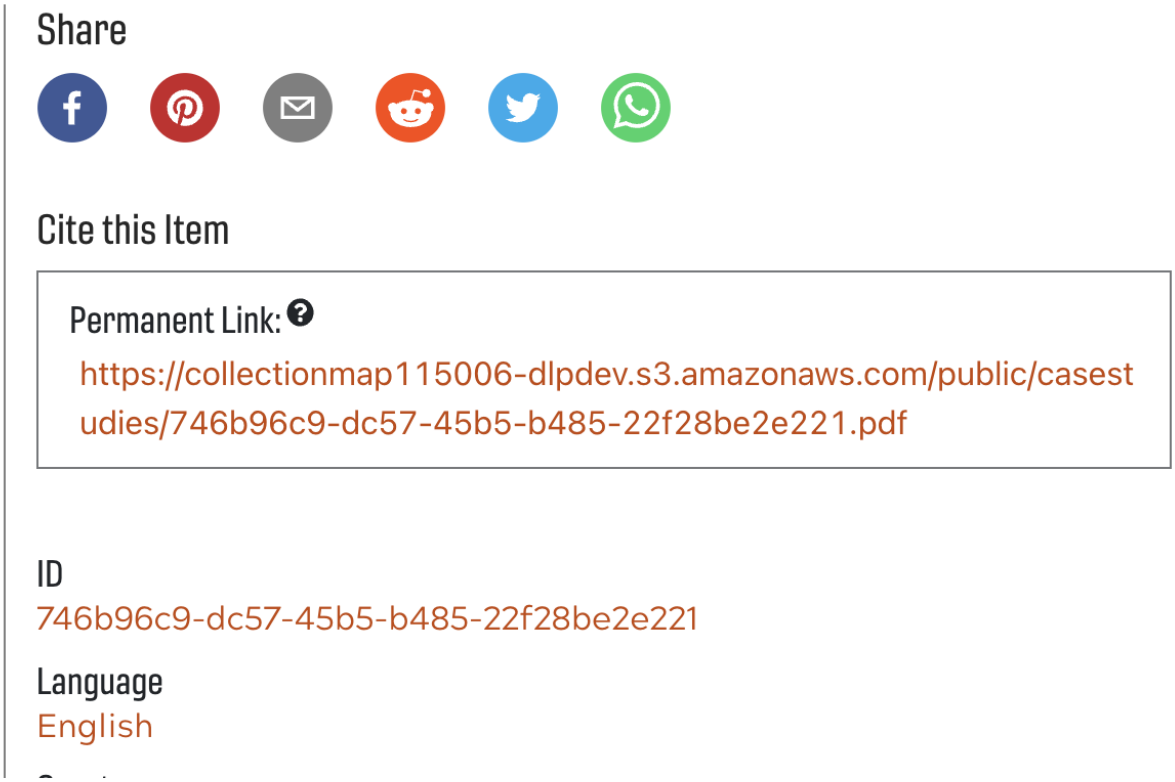

Figure 11: Case Study ID

7.4.4 Edit Metadata

Once confirmed, there will be numerous metadata fields that can be updated. Along with this, there is also an option to delete. If you would like to delete the case study, refer to section 7.4.5. To edit, select the “Edit” option to the side of the “Current mode:” label. For this implementation, the only relevant fields to edit are the collection, description, title, and creator. Once done editing the form, it is necessary to input the collection name of the case study. This is done under the field titled “Collection (by identifier)”. The collection can be found in the path of the case study when it is displayed, as shown in Figure 12 with the case study being a part of the “Internet (ICT)” collection. Once finished, select the “Update Item Metadata” button at the bottom of the page.

Figure 12: Collection identifier of case study

7.4.5 Delete Case Study

Another option is to delete the case study completely. This involves going through the steps to edit the case study metadata, but instead of selecting “Update Item Metadata”, an administrator should choose the “Delete Item” button. This will prompt a redundant confirmation message on which the user should select “ok”. This will delete the case study from being presented on the site, but the case study file itself will remain in the persistent storage on the backend.

8.0 Developer’s Manual

The implementation of the case study library is heavily dependent upon an understanding of the underlying principles of the AWS serverless architecture [31] and React [11]. While much of the site itself can be edited and modified through the site admin page, there is also much to be said for adding custom functionality that does not fit within the scope of the existing DLP Access [8] architecture.

8.1 The React Frontend

8.1.1 Pulling from GitHub

To modify the current instance beyond the capabilities of the site admin page, a developer should begin by obtaining credentials to the AWS backend of the site instance. This gives access to the AWS console containing much of the backend functionality as well as the hosting service, AWS Amplify [13].

Following this, a developer should clone the repository of the most recent case study library GitHub [21]. Contribution access to this repository can be granted upon request. This GitHub link can be found by going to the AWS console and selecting Amplify. Then, by selecting the “casestudies” app on the “All Apps” page, the developer should be led to an interface similar to Figure 13.

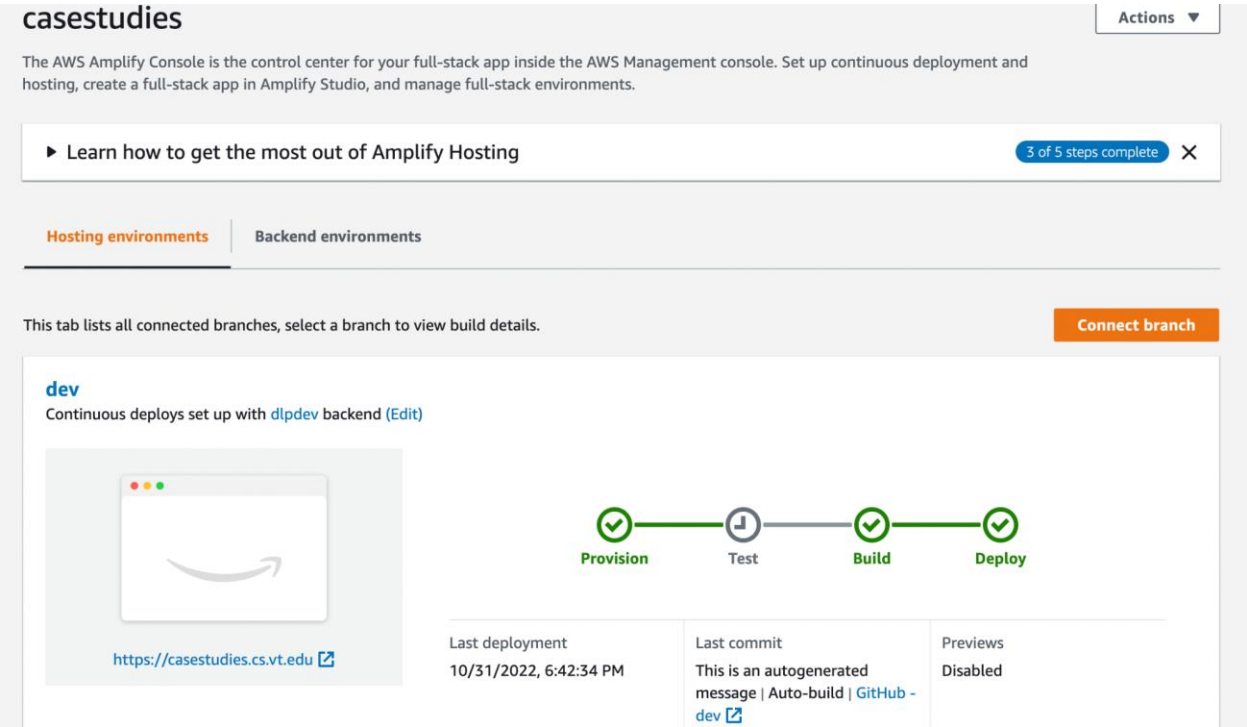

Figure 13: AWS Amplify console

Note how there are two different tabs, one for the hosting environment and one for the backend environment. By selecting the “Hosting environments” tab, a developer can view the different GitHub branches currently deployed. In this instance, the dev branch is the production branch that is routed to the public URL.

There may be more than one branch that is currently deployed. If so, they will be routed to a different URL for development purposes.

To create a new deployment, create a new branch in project GitHub. Once this has been done, navigate back to AWS Amplify and click the orange “Connect branch” button in Figure 13. This will lead to a page similar to the one shown in Figure 14.

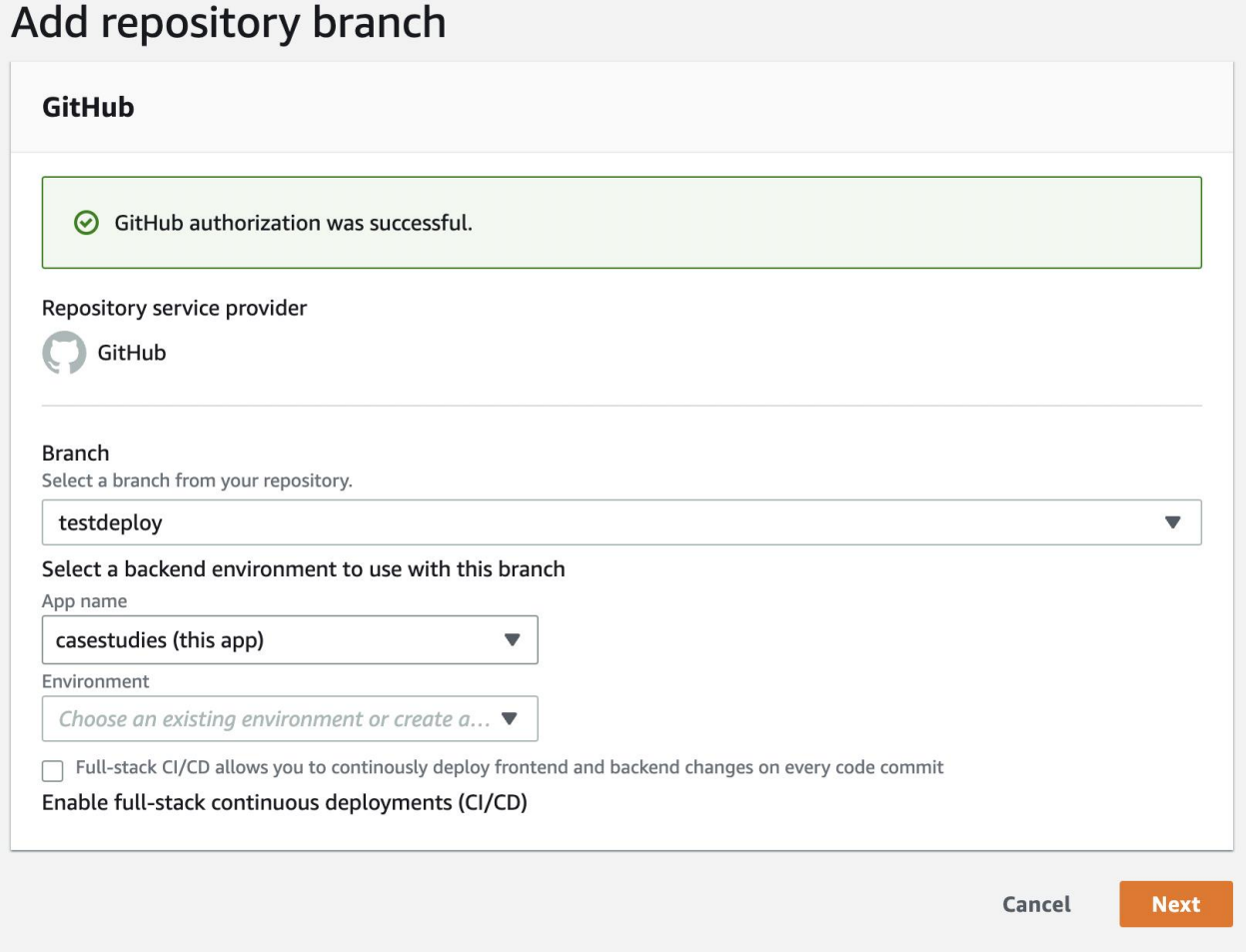

Figure 14: AWS Amplify connect branch page

Choose the branch you would like to deploy from the dropdown, leave the app name as the default, and choose the backend environment in the dropdown with the label “Environment”. This should be set to “dlpdev”. Mark the checkbox to enable full stack continuous deployment. Clicking next will navigate to a confirmation page. Click “save and deploy”. Now, you should see your branch name under the “Hosting environments” tab, where it should be building itself.

Once it has completed the provisioning, building, and deploying processes, click the URL to see the frontend of your branch. Any time a developer commits code to that branch, it will begin the build process over again.

8.1.2 React Frontend Project Structure

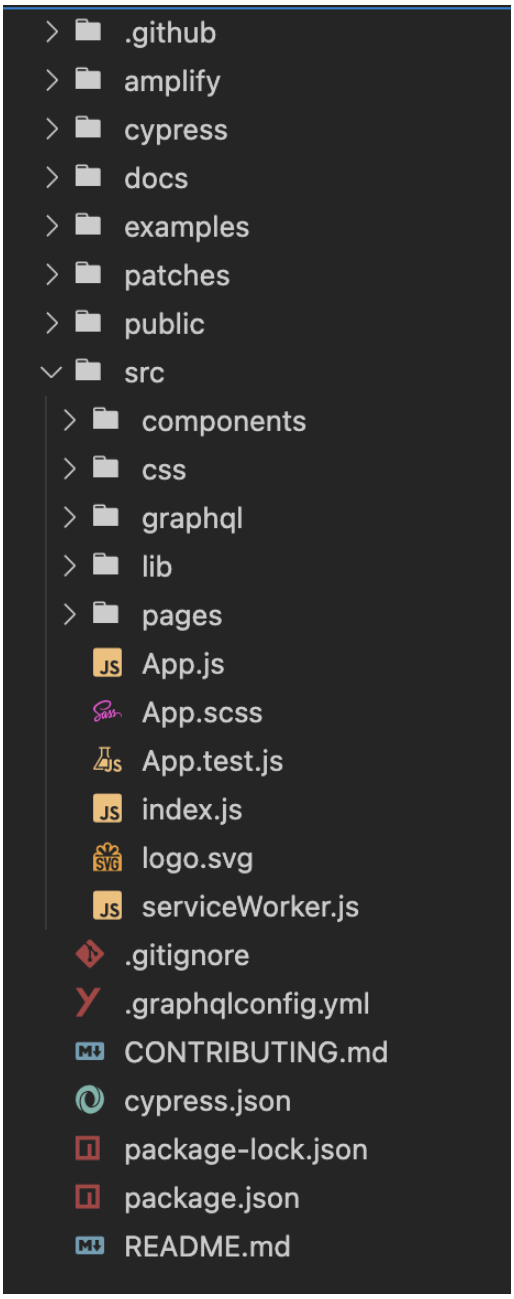

Figure 15: Files and folders of React frontend

Figure 15 shows the folder structure of the library's frontend. The ./src directory is where most relevant files are located. The pages folder holds files representing different pages of the web application, the components folder holds the React components that make up the pages, and the css folder holds the .scss [9] files for styling the pages and components.

8.1.3 Relevant File Inventory

The following is an inventory of the files that were changed in this semester's iteration of the Case Study Library, shown in Table 1.

Table 1: Relevant Files of React App

Filename	Path From Project Root	Description/Comments
CaseStudyUploadPage.js	cs3604/src/pages/CaseStudyUploadPage.js	React component used as the base component of the student upload page.
UploadSection.js	cs3604/src/components/UploadSection.js	Component which renders the form for student upload. This handles the metadata upload as well as the S3 file upload.
adminForms.scss	cs3604/src/css/adminForms.scss	Style file used for both admin forms as well as the upload section component.
CustomPageRoutes.js	cs3604/src/lib/CustomPageRoutes.js	File in which new custom pages need to be imported to. This gives the site admin page access to display the file.
Citation.js	cs3604/src/components/Citation.js	File which displays the permanent link. Changed to display S3 bucket link.
MetadataRenderer.js	cs3604/src/lib/MetadataRenderer.js	File which displays the metadata for a case study (beneath case study on site). This was changed to display the associated file link.

8.1.4 Adding a Custom Page to the Frontend

Adding a custom page to the frontend of the Case Study Library involves not only pushing changes to the GitHub repository, but also making changes to the site via the site admin page [22].

Take, for example, the student upload page that was added this semester shown in Figure 16. Within this page is the component which renders the content of the page itself, UploadSection. Note how it is by default exporting the class CaseStudyUploadPage.

src > pages > JS CaseStudyUploadPage.js > CaseStudyUploadPage > render

dsetareh, 4 weeks ago | 3 authors (You and others)

```
1  import React, { Component } from "react";
2  import SiteTitle from "../components/SiteTitle";
3  import UploadSection from "../components/UploadSection";
4  import { getFile } from "../lib/fetchTools";
5
6  import "../css/TermsPage.scss";
   dsetareh, 4 weeks ago | 2 authors (You and others)
7  class CaseStudyUploadPage extends Component {
8    constructor(props) {
9      super(props);
10     this.state = {
11       copy: ""
12     };
13   }
14
15   componentDidMount() {
16     const htmlUrl = JSON.parse(this.props.site.sitePages)[this.props.parentKey]
17       .data_url;
18     getFile(htmlUrl, "html", this);
19   }
20
21   render() {
22     return (
23       <>
24         <div className="row terms-page-wrapper">
25           <div className="col-12 terms-heading">
26             <SiteTitle
27               siteTitle={this.props.site.siteTitle}
28               pageTitle="Permissions"
29             />
30             <h1 id="permissions-heading">Case Study Student Upload</h1>
31           </div>
32           <div className="col-12 upload-section">
33             <UploadSection />
34           </div>
35         </div>
36       </>
37     );
38   }      You, last month • upload page
39 }
40
41 export default CaseStudyUploadPage;
42
```

Figure 16: Case study upload page component

Once a page has been created, a developer should navigate to the CustomPageRoutes.js file and edit it. Figure 17 shows what the outcome of these edits would look like for the CaseStudyUploadPage within the CustomPageRoutes.js file. Note lines 6 and 10 of Figure 17.

```
1  import React from "react";      Lee Hunter, 2 years ago • begin making routes c
2  import { Route } from "react-router-dom";
3  import AboutPage from "../pages/AboutPage";
4  import PermissionsPage from "../pages/PermissionsPage";
5  import AdditionalPages from "../pages/AdditionalPages";
6  import CaseStudyUploadPage from "../pages/CaseStudyUploadPage";
7  const pageComponents = {
8    AboutPage: AboutPage,
9    PermissionsPage: PermissionsPage,
10   CaseStudyUploadPage: CaseStudyUploadPage,
11   AdditionalPages: AdditionalPages
12 };
13
14 function route(site, key, path, PageComponent, childKey = null) {
15   let elemKey = key;
16   if (childKey) {
17     elemKey += "." + childKey;
18   }
19   return (
20     <Route
21       key={elemKey}
22       path={path}
23       exact
24       render={props => (
25         <PageComponent site={site} parentKey={key} childKey={childKey} />
26       )}
27     />
28   );
29 }
```

Figure 17: CustomPageRoutes.js component edits for CaseStudyUploadPage

Once the CustomPageRoutes.js file is configured correctly for the new page, navigate to https://casestudies.cs.vt.edu/siteadmin. This page requires authentication as it provides site editing functionality. If you are working on this project in a future semester, please reach out to the client to retrieve these credentials.

Once logged in, navigate to the Site Pages Config tab shown in Figure 18. Change the current mode to “Edit” and update data accordingly based upon the component name you wish to display. Adding a new page will involve adding a new page via the “new page” button in the bottom right-hand corner of the edit window, then selecting “update site” once finished. For the “Component” field, make sure the component name is the same component name declared in the pageComponents object in Figure 17.

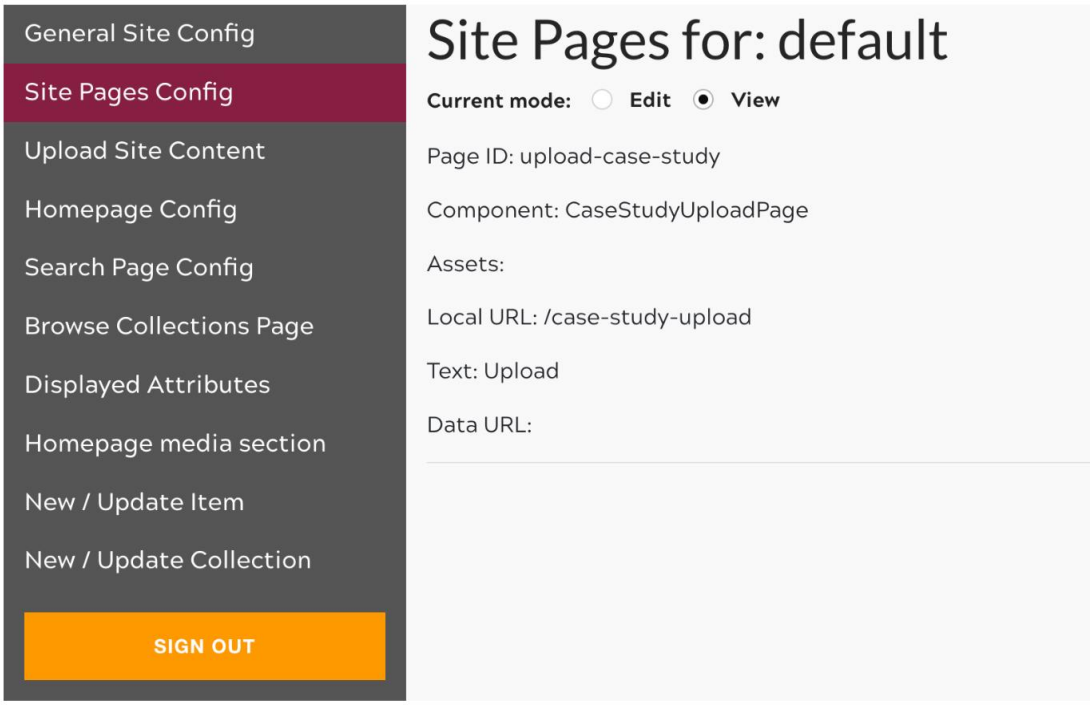

Figure 18: Site Pages Config tab on siteadmin page

Once updated, the site should now show a button to navigate to the new page in the top nav bar on wide windows, and in the side nav menu on smaller width windows. This button will be labeled by what a developer enters into the “Link Text” field while editing Site Pages Config.

8.2 AWS Infrastructure

If you are working on this project in a future semester, contact the project client for credentials. Once these credentials have been obtained, log into the AWS Console [32] at https://aws.amazon.com/console/.

8.2.1 Relevant AWS Resources

Shown in Table 2 are the resources used in the development of the Case Study Library.

Table 2: Relevant AWS Resources

AWS Service	Name	Description/Comments
Amplify [13]	casestudies, hosting: dev, backend: dlpdev	The deployment platform the Case Study Library uses. Gives easy front-end access to S3, dynamoDB, and authentication functionality. You can find the services used through Amplify via the backend environment.
Cognito [18]	iawav2658176f3_userpool_658176f3-dlpdev	Provides authentication functionality for siteadmin page and the case study upload page. Does not allow access to DynamoDB write functionality unless the user group is correct.
DynamoDB [16]	Archive-dcfeiwidx5ezjiipllgulq45ue-dlpdev, Collection-dcfeiwidx5ezjiipllgulq45ue-dlpdev, Site-dcfeiwidx5ezjiipllgulq45ue-dlpdev	Stores the metadata for the collections, archives, and site configuration data. The S3 link to the case studies and their thumbnails are stored here.
S3 bucket [12]	collectionmap115006-dlpdev, public/casestudies/*: case study files, public/sitecontend/image/default/*: thumbnail files and other site files	Stores files of case studies, thumbnails for them and the collections, and other site graphics
AppSync [33]	collectionarchives-dlpdev	GraphQL API used for

		querying metadata.
Lambda [26]	thumbnail-CreateThumbnail-BVxDrZJvihPE	Serverless function triggered by case study file uploads to the S3 bucket. Creates a thumbnail for the PDF or MP4 file and uploads the small PNG file to the thumbnail file folder of the S3 bucket.

8.2.2 Important Maintenance Notes

The GraphQL API key is set with an expiration date. For security purposes, this API key expires at most a year after the expiration date is set. This is the API key used for authentication to the API. If it expires, the site will go down. To reset the expiration date, navigate to AWS AppSync > Collectionarchives-dlpdev > Settings > API key, select the API key and select “Edit”. Change the expiration date, as shown in Figure 19, and hit “Save”.

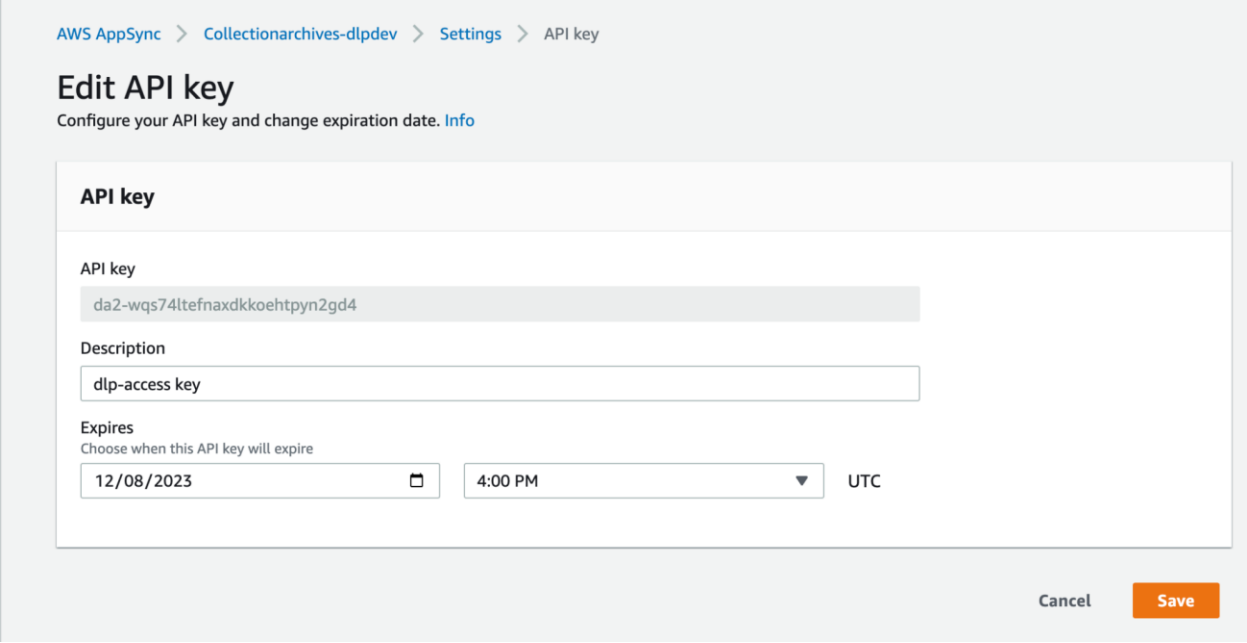

Figure 19: Editing the GraphQL API key

9.0 Lessons Learned

This project involved working with a codebase that utilized cloud services heavily. This meant that there was a steep learning curve for group members that did not have much experience with those technologies. It was found that to produce solutions on preexisting systems, it is necessary to ask for help when stuck. This especially holds true for the times when a solid understanding of the architecture is crucial for the development of a solution. Since we had to modify many aspects of the services and codebase, it would have been helpful to ask a maintainer of the front-end architecture for guidance early in the development process.

Communication is a large part of any project. During the development of the site additions and modifications, some members found that the same issues were being addressed at the same time. This led to some redundant work having to be thrown out. Overall, this was a great learning experience. It helped members learn to communicate better by notifying others as to what they were working on before work started. This made for a more lean and efficient development process.

9.1 Timeline/Schedule

The following are two tables with information about our work completed, problems identified, and solutions identified through this semester. Table 3 outlines September and October, while Table 4 outlines November and December.

Table 3: Project Timeline (September - October)

	September	**October**
Completed	• AWS [3] Credentials obtained • Bi-Weekly meeting time established and scheduled with Professor Dunlap • Met with Professor Dunlap to ask introductory questions about the Library	• Admin Page Credentials obtained • Modifiable site data from site Admin page changed • Re-Deployment of site attempted • Site costs audited

	September	October
	• Case Study Upload page wireframe designed	
Problems Identified	• Obtaining Site Admin Page Credentials • Previous implementation to add case studies to the current library is very obtuse • Many broken images on the site currently • Searching functionality fully broken • Full site audit requested, both out-of-date packages and running costs	• Case Studies are not easily browsable by course topic, despite the categories existing • Expired GraphQL API Key [17] has caused it to go down • Previously uploaded Case Studies have disappeared from the library webpage
Solutions Identified	• Reworking submission page • Ask prior group for site • Admin Credentials • General site cleanup	• Bringing the Case Study Library site back online • Retrieval and addition of prior case studies

Table 4: Project Timeline (November - December)

	November	**December**
Completed	• Site brought back online • Case Study Topic Categories created • Case Study Categorical filtering and search implemented • Continuation of AWS DynamoDB [16] Updates	• Report finalized • Offboarding meeting with Professor Dunlap
Problems Identified	• API Key used to populate site info was out of date due to a short expiry time • Previously uploaded case studies are missing categorical information related to Course Topic • Authentication is performed through Instructor-Created accounts generated per semester, rather than VT SLO • Site can only display and accept PDF files	

	November	December
Solutions Identified	● API Key renewed for its maximum allowed length (1 Year) ● Harnessing the efforts of VT Honor students to perform categorization on prior Case Studies ● Investigation of the use of VT SLO for Case Study Library Authentication	

9.2 Problems

Despite the large amount of progress made towards site improvements by our team over the course of the semester, many problems were encountered that hampered our ability to develop and improve site functionality.

Our most pressing problem which nearly persisted throughout the first half of the semester was the site becoming inaccessible. Halfway through implementation of our improvements, the site ceased to function, displaying only an unending loading icon as shown in Figure 20.

Loading

Figure 20: Message displayed during site outage

Our team went to great lengths to understand what had exactly gone wrong to cause the Case Study Library to go down in such a silent fashion. Much of our effort initially went towards reverting our own various changes to both the codebase and the AWS settings and architecture to determine which modifications had caused the issue. Regardless, work continued to be performed, albeit at a much slower pace due to the complete inability to test against our production backend.

9.3 Solutions

Just as problems were encountered throughout the project, solutions were weighed against another, discussed amongst the team, and implemented upon unanimous approval. Many of our issues were solved through team brainstorming, and every member of the team has helped solve at least one problem that we encountered. However, as the most pressing problem, the solution to the Case Study Library's downtime was the most helpful.

After much investigation, it was finally determined that our modifications had no correlation with the Case Study Library downtime. What was at fault was the backend API Key generated by the prior team, which had expired during our process of acclimating ourselves with the environment. Once the key was renewed (this time for its maximum allowed length) the site returned online, and live development could be performed once again.

9.4 Future Work

We have several tasks laid out that can be accomplished by future teams working on the Case Study Library.

VT CAS Authentication for certain aspects of the Case Study Library is an important next step to finishing the site and making it sustainable. Currently, the site authenticates with a password-like approach, in which students are given account credentials backed by AWS Cognito User Pools to access the upload page. However, after a student uploads a case study, they cannot alter or delete it, and having such functionality that would be reliable requires the student to be authenticated through Virginia Tech. Similarly, this would allow for more security for the backend resources of the site, given that any misuse of the upload feature could be tracked down and potentially be an honor code violation [34]. In order to do this, a new metadata tag would need to be added to the archives that associates the archive with the PID of the Virginia Tech student. Also, a solution would need to be derived to either allow for the searching of a given archive where it can be edited or deleted following authentication or allow for this directly on the archive page. It should be noted that some case studies have an associated file where two library archives are the same case study, so this functionality should likely apply to both archives. Furthermore, existing case studies should likely be excluded from this functionality.

Next, one of the most important updates that can be made is a cheaper alternative to using AWS OpenSearch [28]. In the current implementation, OpenSearch uses an ElasticSearch instance that is charged hourly to facilitate GraphQL queries, mutations, and deletions. However, this rate is being charged regardless of the amount of traffic the site has, in which a solution that utilizes a pay-per-use kind of search could greatly reduce the cost of the site. While there are several solutions that could be implemented to reduce this cost, any future team should consult with Dr. Chen, a Virginia Tech AWS Solution Expert, to find the best solution that can be implemented in the Case Study Library's case.

Finally, in our final meeting with Professor Dunlap, he suggested an interest in having a living and evolving element to the library, which could give the library aspects of a platform, instead of just a repository. While the previous two updates should have a higher priority over this, there is potential for allowing comments on case studies once authentication is properly implemented. Also, other ideas include a ranking system for case studies or even a forum for case study ideas or proposals. When thinking about how these ideas could be implemented into the site, there will likely be changes needed to the backend functionality as well as the frontend, so consulting with Dr. Chen to share the implementation ideas the team comes up with will be important.

10.0 Acknowledgements

Professor Dunlap (dunlapd@vt.edu) has been incredibly helpful and gracious throughout our project by not only giving us tips on how to improve our site, but also tips on what could be analyzed and discussed within our final report. Our bi-weekly meetings were incredibly productive and helpful, and we could not have performed our improvements without his help.

Dr. Yinlin Chen (ylchen@vt.edu) has been instrumental in helping us with AWS issues and obtaining access to the initial Case Study Library and source code.

Dr. Edward Fox (fox@vt.edu), our professor for this course, has helped our team keep track and keep up progress towards our project goals through the structure of the class itself.

11.0 References

[1] Virginia Polytechnic Institute and State University, “CS3604: Professionalism in computing,” *Computer Science | Virginia Tech*, 24-Sep-2020. [Online]. Available: https://website.cs.vt.edu/content/website_cs_vt_edu/en/Undergraduate/courses/CS3604.html. [Accessed: 10-Dec-2022].

[2] B. Wieder, K. Papili, S. Jain, K. Bekele, D. Marin, and B. Nguyen, “CS3604 Case Study Library,” Virginia Tech CS4624 team term project submission. *Virginia Polytechnic Institute and State University,* May 2022, [Online]. Available: http://hdl.handle.net/10919/109970. [Accessed: Nov. 15, 2022].

[3] The Python Software Foundation, “Welcome to Python.org,” *Python.org*, 2022. [Online]. Available: https://www.python.org/. [Accessed: 09-Dec-2022].

[4] Amazon Web Services, Inc., “Cloud computing services - amazon web services (AWS),” *Amazon Web Services, Inc.,* 2022. [Online]. Available: https://aws.amazon.com/. [Accessed: 15-Dec-2022].

[5] Virginia Polytechnic Institute and State University, “3604: Professionalism in Computing | Home.” *Virginia Polytechnic Institute and State University*, 2022. Available: https://casestudies.cs.vt.edu/. [Accessed: 09-Dec-2022].

[6] Virginia Polytechnic Institute and State University, “Department of Computer Science | Computer Science | Virginia Tech.” *Virginia Polytechnic Institute and State University,* 2022. Available: https://cs.vt.edu/. [Accessed: 09-Dec-2022].

[7] Y. Chen, “End to end Serverless Digital Repository Platform on the Cloud,” presented at the 17th International Open Repositories Conference, Denver, CO, USA, *Virginia Polytechnic Institute and State University,* 08-Jun-2022. Available: http://hdl.handle.net/10919/111198. [Accessed: 09-Dec-2022].

[8] Y. Chen, L. Hunter, A. Waldren, T. Jiang, P. Mather, “vt-digital-libraries-platform/dlp-access: DLP Access Website: A Multi-Tenancy Serverless web application.” *Github,* 2022. [Online]. Available: https://github.com/vt-digital-libraries-platform/dlp-access. [Accessed: 06-Dec-2022].

[9] Sass, “Sass: Syntactically Awesome Style Sheets.” *Sass,* 2022. [Online]. Available: https://sass-lang.com/. [Accessed: 09-Dec-2022].

[10] Meta Platforms, Inc., “Components and Props – React.” *Meta Platforms, Inc.,* 2022. [Online]. Available: https://reactjs.org/docs/components-and-props.html. [Accessed: 09-Dec-2022].

[11] Meta Platforms, Inc., “React – A JavaScript Library For Building User Interfaces.” *Meta Platforms, Inc.,* 2022. [Online]. Available: https://reactjs.org/. [Accessed: 06-Dec-2022].

[12] Amazon Web Services, Inc., “Cloud Object Storage – Amazon S3 – Amazon Web Services,” *Amazon Web Services, Inc.,* 2022. [Online]. Available: https://aws.amazon.com/s3/. [Accessed: 09-Dec-2022].

[13] M. Hollands, “AWS Amplify – Scalable, Full-Stack Web and Mobile Apps, Built on AWS – Amazon Web Services,” *Amazon Web Services, Inc.* 2022. [Online]. Available: https://aws.amazon.com/amplify/. [Accessed: 06-Dec-2022].

[14] Amazon Web Services, Inc., “Bucket Policies and User Policies - Amazon Simple Storage Service.” *Amazon Web Services, Inc.,* 2022. [Online]. Available: https://docs.aws.amazon.com/AmazonS3/latest/userguide/using-iam-policies.html. [Accessed: 09-Dec-2022].

[15] Amazon Web Services, Inc., “Storage - Getting started - JavaScript - AWS Amplify Docs.” *Amazon Web Services, Inc.,* 2022. [Online]. Available: https://docs.amplify.aws/lib/storage/getting-started/q/platform/js/docs.amplify.aws/lib/storage/getting-started/q/platform/js/. [Accessed: 09-Dec-2022].

[16] D. Rangel, Amazon Web Services, Inc., “Fast NoSQL Key-Value Database – Amazon DynamoDB – Amazon Web Services,” *Amazon Web Services, Inc.,* 2022. [Online]. Available: https://aws.amazon.com/dynamodb/. [Accessed: 09-Dec-2022].

[17] Amazon Web Services, Inc., “API (GraphQL) - Getting started - JavaScript - AWS Amplify Docs.” *Amazon Web Services, Inc.,* 2022. [Online]. Available: https://docs.amplify.aws/lib/graphqlapi/getting-started/q/platform/js/. [Accessed: 15-Dec-2022].

[18] H. Roose, “Customer Identity and Access Management – Amazon Cognito – Amazon Web Services,” *Amazon Web Services, Inc.,* 2022. [Online]. Available: https://aws.amazon.com/cognito/. [Accessed: 09-Dec-2022].

[19] H. Roose, “Amazon Cognito user pools - Amazon Cognito.” *Amazon Web Services, Inc.,* 2022. [Online]. Available: https://docs.aws.amazon.com/cognito/latest/developerguide/cognito-user-identity-pools.html. [Accessed: 09-Dec-2022].

[20] Y. Chen, L. Hunter, S. Ghosh, T. Giang, J. Tuttle, and A. Waldren, "*VTDLP Overview*" *Virginia Polytechnic Institute and State University,* 2022. [Online]. Available: https://about.digital.lib.vt.edu/project/. [Accessed: 09-Dec-2022].

[21] T. Bagbey, M. Betsill, and D. Setareh, "CS3604: DLP access website: A multi-tenancy serverless web application," *GitHub*, 2022. [Online]. Available: https://github.com/bbtrevor/cs3604/tree/dev. [Accessed: 15-Dec-2022].

[22] Y. Chen, L. Hunter, S. Ghosh, T. Giang, J. Tuttle, and A. Waldren, *"Site Admin Page." Virginia Polytechnic Institute and State University,* 2022. [Online]. Available: https://casestudies.cs.vt.edu/siteadmin. [Accessed: 09-Dec-2022].

[23] Javascript.com, "JavaScript.com." 2022. [Online]. Available: https://www.javascript.com/. [Accessed: 09-Dec-2022].

[24] Ant-Design, "Ant-Design/ANT-design: An enterprise-class UI design language and react UI Library," GitHub - ANT-design. 2022 [Online]. Available: https://github.com/ant-design/ant-design. [Accessed: 09-Dec-2022].

[25] The GraphQL Foundation, "Queries and mutations," *GraphQL*, 2022. [Online]. Available: https://graphql.org/learn/queries/. [Accessed: 15-Dec-2022].

[26] D. Musgrave, "AWS Lambda," *Amazon Web Services, Inc.*, 2022. [Online]. Available: https://aws.amazon.com/lambda/. [Accessed: 09-Dec-2022].

[27] C. S. Forester, "Amazon Resource Names (ARNs)," *Amazon Web Services, Inc.*, 2015. [Online]. Available: https://docs.aws.amazon.com/general/latest/gr/aws-arns-and-namespaces.html. [Accessed: 09-Dec-2022].

[28] Amazon Web Services, Inc., "Amazon OpenSearch Service," *Amazon Web Services, Inc.,* 2022. [Online]. Available: https://aws.amazon.com/opensearch-service/. [Accessed: 09-Dec-2022].

[29] Elasticsearch B.V., "Free and open search: The creators of Elasticsearch, Elk & Kibana," *Elastic*, 2022. [Online]. Available: https://www.elastic.co/. [Accessed: 09-Dec-2022].

[30] Virginia Polytechnic Institute and State University, "Honor code policy and Manual," *Office of Undergraduate Academic Integrity* | *Virginia Tech*, 26-May-2022. [Online]. Available:

https://honorsystem.vt.edu/content/honorsystem_vt_edu/en/honor_code_policy_test.html. [Accessed: 09-Dec-2022].

[31] D. Musgrave, “Building Applications with Serverless Architectures,” *Amazon Web Services, Inc.*, 2022. [Online]. Available: https://aws.amazon.com/lambda/serverless-architectures-learn-more/. [Accessed: 15-Dec-2022].

[32] M. Vichi, “AWS Management Console,” *Amazon Web Services, Inc.*, 2015. [Online]. Available: https://aws.amazon.com/console/. [Accessed: 15-Dec-2022].

[33] A. Patel, “AWS AppSync,” *Amazon Web Services, Inc.*, 2015. [Online]. Available: https://aws.amazon.com/appsync/. [Accessed: 15-Dec-2022].

[34] Virginia Polytechnic Institute and State University, “Login Service CAS Protocol Integration,” *Login Service CAS Protocol Integration | Middleware Services*, 2022. [Online]. Available: https://www.middleware.vt.edu/sso/cas.html. [Accessed: 15-Dec-2022].