Critical Design Review:
Web Interface for Mercer University

Kaleb Kushinka

Martin Ramos

Client: Bremen Vance, PhD

Project Manager: Stephen Hill, PhD

April 10, 2023

Table of Contents

List of Figures

Executive Summary

The School of Engineering at Mercer University implements multiple research projects for students to utilize skills and knowledge learned through classes. In hopes that the students receive necessary experience, leading them to success in their specified field of study. The projects require the students to research numerous information sources and submit documentation to show full understanding of the task assigned or chosen. Thousands of students tackle these projects every year in retrospect to Mercer's pristine reputation of student success. Due to the enormous inflow of documentation, the research can be hard to filter through and find. The documents are stored within physical binders or massive files within a local computer system. Project documents withhold valuable research material and inspiration for future students and faculty.

An online web interface containing the project documents in a catalog fashion, allowing for access to requested information through filters and search features. The interface would allow the user to visually read the documents through a PDF viewer, while also containing accessibility features to adhere to certain individual needs. The project came by the request of Dr. Bremen Vance, a technical communication professor at Mercer University. The initial request was for a digital archive tool with the possibility to search for specific documents and view them in an organized structure. It also needed to avoid the use of subscription based programs to prevent rolling charges to the university. The features use a back-end database to support their functionality. The database stores the metadata used for filtering out the specific request made by the user. A technical communication Mercer student, participating in an independent study, was responsible for scanning the project documents into the computer system. The student then manually inputted the metadata for each individual project into an excel sheet. A sample of technical communication project documentation was used for the initial implementation of the interface.

The team consisted of two computer engineering majors, Kaleb Kushinka and Martin Ramos.The team possessed the required skill set to qualify for the goals and requests made by the client. They are in their fourth year within Mercer University's computer engineering program, which masterfully prepares students with the necessary knowledge in multiple programming languages and computer networks. The students have also participated in long term internships, providing significant experience in the required knowledge to complete the requested task. This includes experience with windows applications, web interfaces, and database management. Visual Code and MySQL were used for coding and databases. The code and documentation will be shared via GitHub and Google Drive. The project milestones were scanning document metadata, developing the web interface back-end, incorporating the metadata within the back-end functionality, designing the front-end, and allowing room for future development.

Acronyms

API - application programming interface

CDR - critical design review.

PDF - portable document format.

PDR - preliminary design review.

SQL - structured query language.

WWW - world wide web.

1 Introduction

Mercer's School of Engineering produces a massive amount of documentation from technical communication to engineering senior design projects. These projects help prepare students with real hands-on experience within their field of study. Faculty and students use these documents for personal research purposes or inspiration brainstorming new project ideas for upcoming projects. The current system of storage is mainly physical binders and large files stored in local computer system memory. The system is not only unsecure, but also unorganized making it impossible to pull specific documents from the archive. This reduces the amount of information available to the faculty and students.

Our client, Dr. Bremen Vance requested the development of an online web interface archiving the documentation from the multiple types of projects. The documents can then be pulled from by features such as search, browse, upload, and the PDF viewer. The interface will create an organized catalog for students and faculty to easily access valuable information. This type of interface is commonly used by non-profit or profitable companies allowing public access to multiple reading material, such as libraries or bookstores. The interface will use back-end databases and features to support a smooth user experience when operating through the webpage. The primary goal of the project was to develop an interface with back-end databases and features with a smooth front-end experience for the user to access requested documentation. Adhering the individual needs through appropriate accessibility and allowing for future development were also important throughout the development process.

The remainder of the report is organized as follows. A summary of the previous PDR, such as goals and design criteria, in Section 2. The final work accomplished, whether that be a minimized or final version, in Section 3. Results and discussion of the project including data, such as test results and other factors, in Section 4. A finalized summary of the project work completed, how the project was set up for future development, and recommendations for future features in Section 5. The references used throughout the report will be apparent in Section 6.

2 PDR Summary

The goal of this project is to design an easy-to-use web interface that will retrieve projects for individuals, such as professors and other trusted persons, in the Mercer School of Engineering. The project archive will allow users to search, filter, and explore varieties of past projects. A professor or student should be able to find projects of their interest in whatever major they teach or study. A successful solution must be a computer program that is easy to navigate and can find any desired project.

The project archive's usability and design will be crucial regarding the merit criteria. When users engage with web interfaces, they hope for a superb user experience. To ensure an efficient user experience, developers must prioritize the individuals' needs, especially simplicity. Additionally, the project archive's design aesthetics must be visually appealing and user-friendly. These features can be determined using the W3C

web standards that are widely used across the World Wide Web (WWW). The W3C standards define an “Open Web Platform” for application development, which come in the form of accessibility (*Standards* 2021).The solution should have widespread adoption and offer customizable features to accommodate desired designs.

The project solution needs to be both affordable, costing less than $1000, and subject to inspection to ensure that it can be developed using the available tools and expertise. It must also meet all legal regulations and laws and be operational, taking into account the procedures and actions necessary for maintenance. Additionally, the project archive should be regarded as an ongoing concept that can be modified by other web designers. Lastly, the solution should be timely and practical, with reasonable deadlines and meetings for the final product.

The team consists of two senior computer engineers, Kaleb Kushinka and Martin Ramos. They will share the responsibility of coding the web interface, while also focusing on the features they have more coding experience. Kaleb will assist more with the back-end databases and features, while Martin will be able to assist more with front-end design. This will also allow for the students to learn more about the areas of coding they lack experience.

As mentioned before, the desired design must be a user-friendly and appealing design. The idea and functions have remained the same since the PDR presentation. However, there have been specific elements of the front-end development that Dr. Gallagher, Dr. Brewer, and Dr.Hill recommend that should be changed. In the upcoming section, prototypes designed by Kaleb and Martin will be displayed showing the progress and changes that were made throughout the design process.

3 Work Accomplished

During the semester, the online web interface had to prioritize core features to deliver a finished product with working user interaction. There was initially a prototype designed through Figma, a prototyping application that allows developers to test out designs before beginning actual coding. The first design of the interface can be seen in Figure 1.

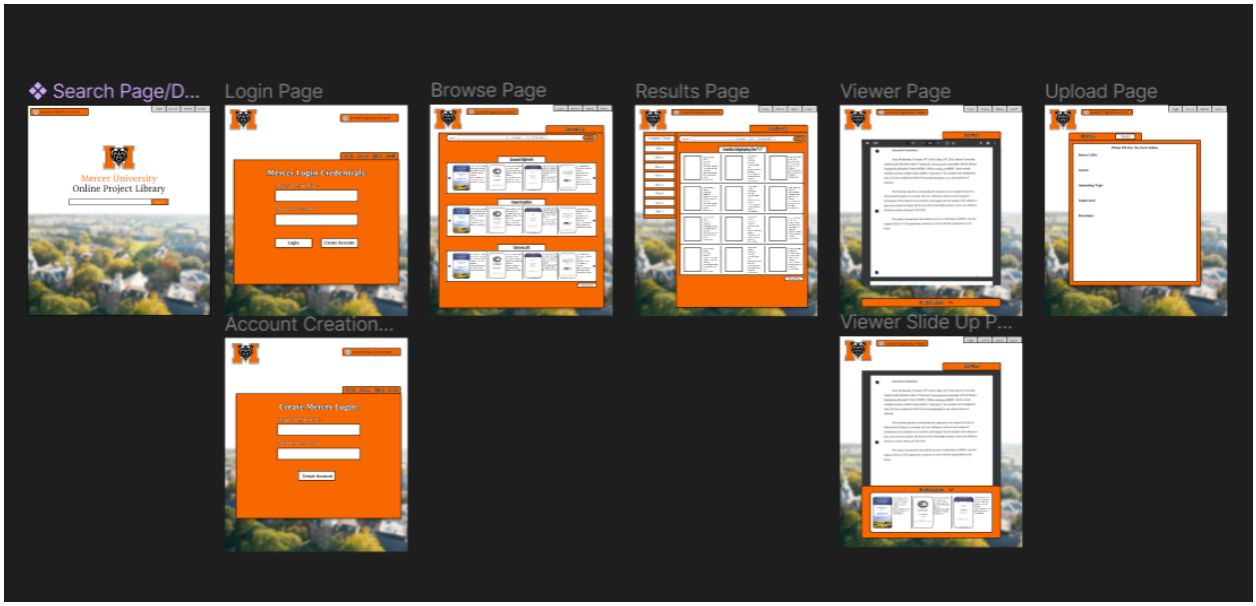

Figure 1: Prototype Design for the First Peer Review.

The peer review produced very valuable feedback on the design of the interface to provide the best user experience possible. Some of the feedback received was that there was reiteration of buttons in the top right corner of the webpages (i.e. browse button on the browse page), a more modern graphical design desired, simplify the filtering system to more common factors like date range, and reduce the amount of results shown in the browse page to not overwhelm the user. The feedback was used to redesign the interface to adhere to the requested material, which can be seen in Figure 2.

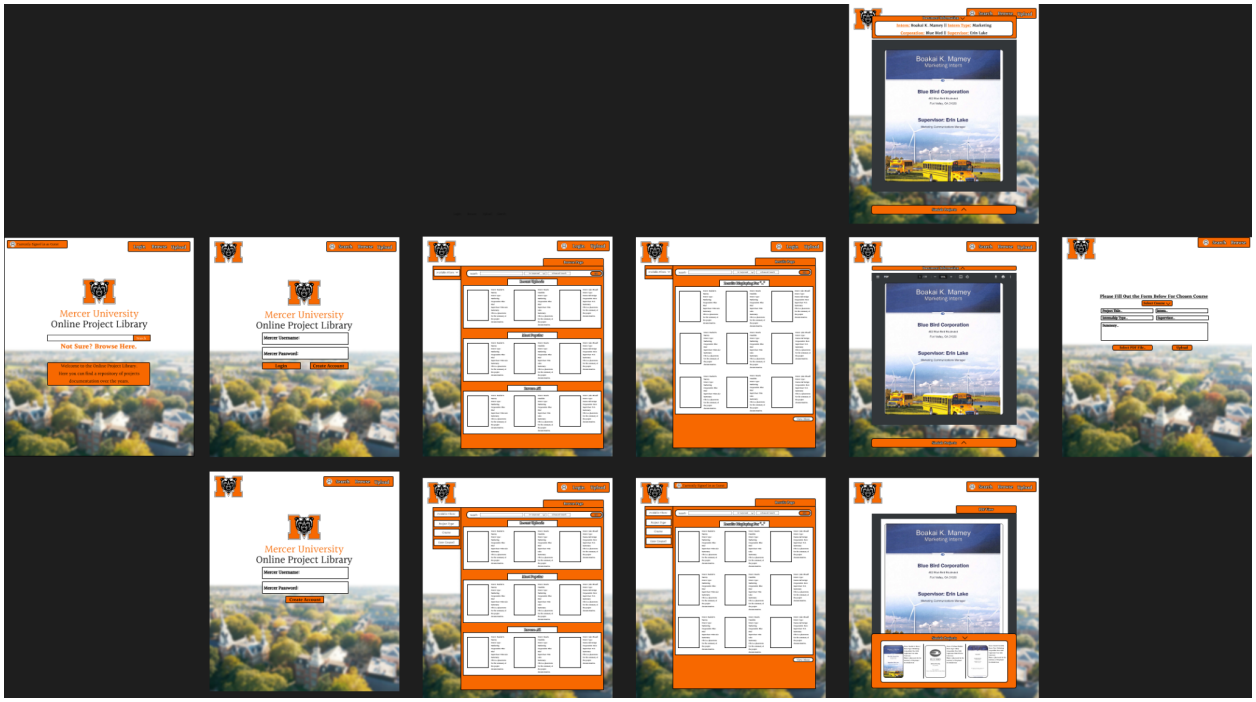

Figure 2: Revised Prototype Design After Feedback.

Due to time constraints of the semester and deliverables, certain features were prioritized. The team and client agreed on making the upload page, browse page, PDF viewer page, and home page with their respective features to be the final deliverables. The team implemented future proofing to the coding, within Visual Code, for the login feature, advanced search function, similar project recommendations, and brief summary automated by an initial analysis of the document uploaded.

The final design of the interface brings the user to an initial home page with router buttons, sending the user to different pages, a search bar to begin searching using keywords, and then a brief summary of the purpose of the project. The home page can be seen in Figure 3.

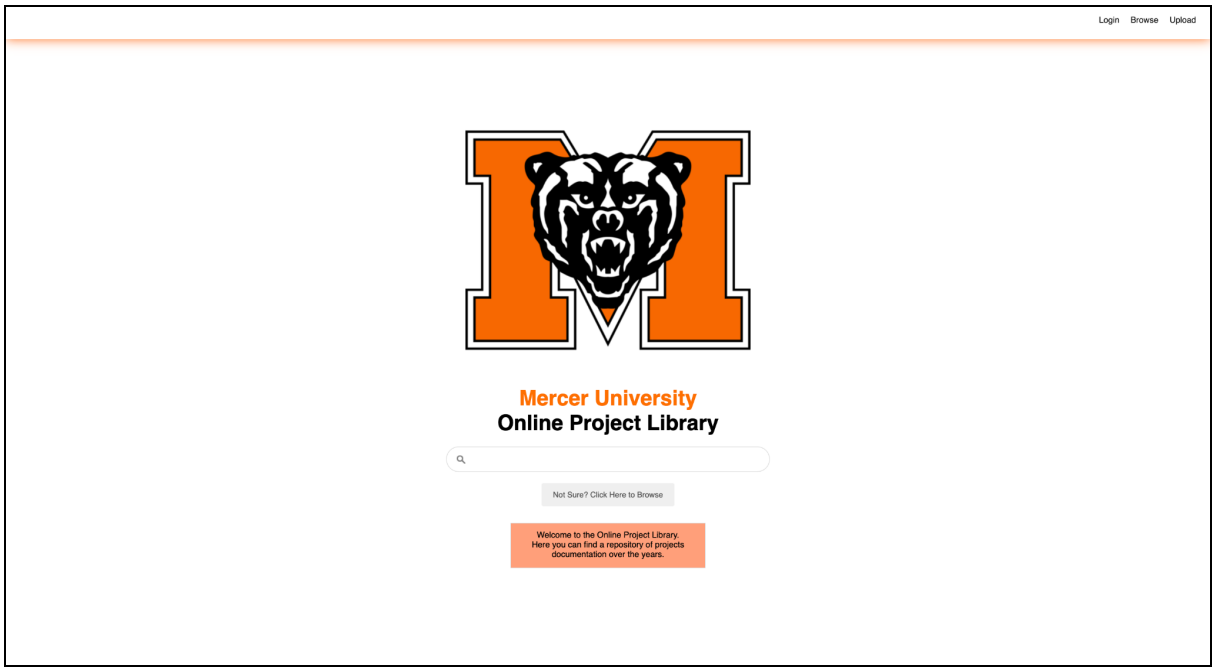

Figure 3: Home Page of the Interface.

The browse page can be accessed via the browse button or by searching a keyword in the search bar on the home page. The browse page hosts a section on the left side of the screen with common filters such as date range, project type, and project manager. The information used to filter the results comes from the metadata stored within the back-end database in MySQL. The main goal of the results being displayed was to minimize the overwhelming effect of some search results systems. They will be displayed in grid stature with adaptability to the web page size and a brief summary of key elements of the document. There will also be an accompanied search bar at the top of the results display for the user to continue searching for specific documentation. The browse page can be seen in Figure 4.

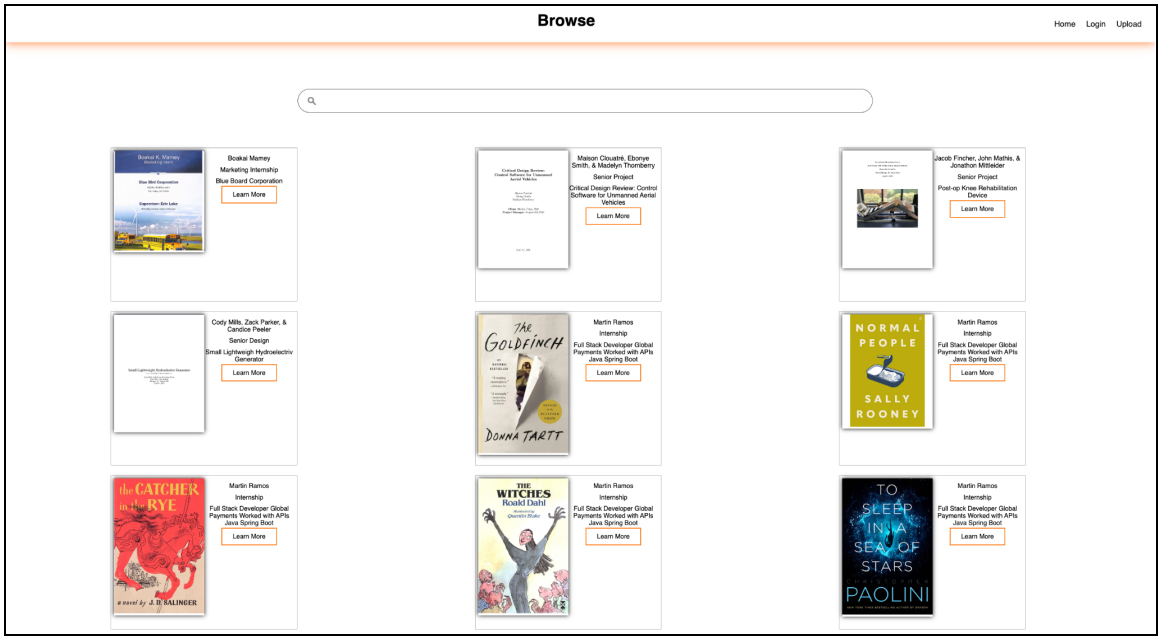

Figure 4: Browse Page of the Interface.

The upload page allows for the user to submit projects into the database to then be pulled from by other users. They are prompted with a form to attach a PDF file and fill out project information to determine the metadata for that specific project. There is a validation system set in place to check if the project being submitted has already been uploaded. It notifies the user respectively. The page can be accessed via the upload button located in the top right corner of the pages. The design of the upload page can be seen in Figure 5.

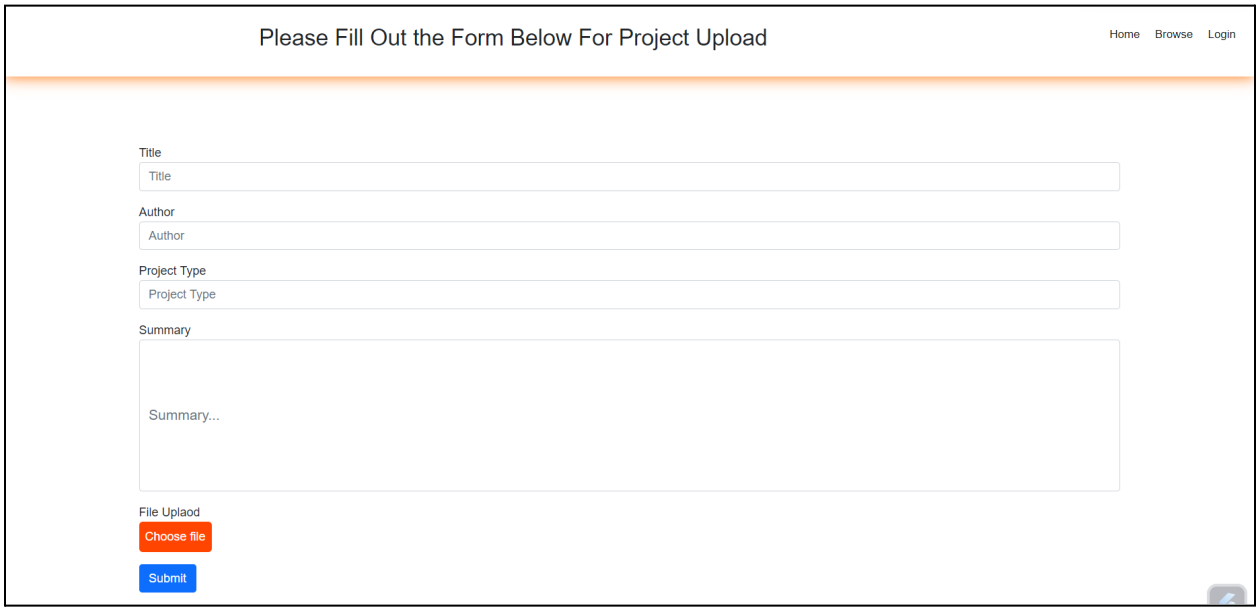

Figure 5: Upload Page of the Interface.

A database allows programmers to store large volumes of data and test the function of the design. In this case, a series of rows and columns were made to store data such as “author” and “project_name”. After connecting the database to the FastAPI framework, tests were done to ensure the framework and database were connected. One test included a GET request to display the data from the database. An example of what the database looks like can be seen in Figure 6.

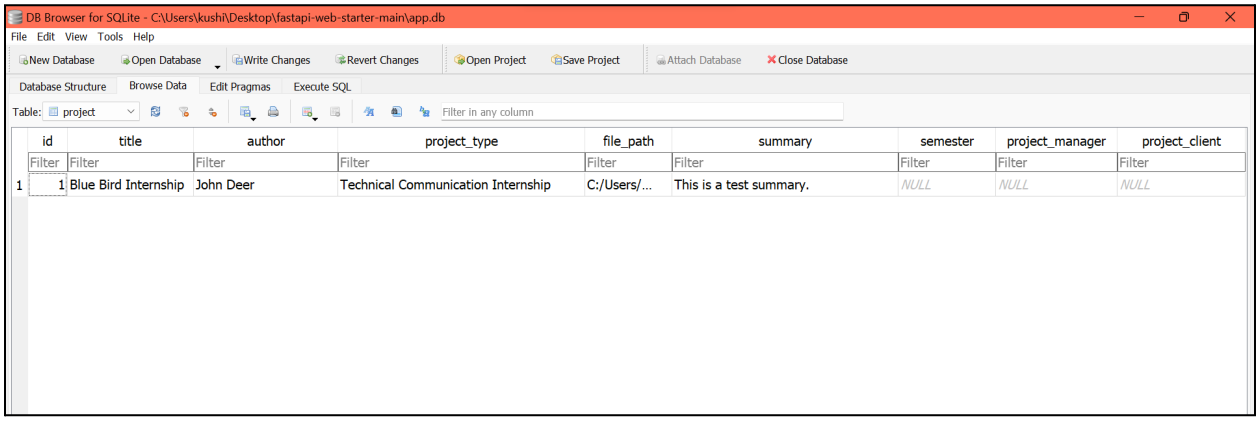

Figure 6: Example of Project in Database.

The PDF viewer page allows for the user to read through the project documents, with accessibility features such as keyboard control. The viewer chosen for the final design of the page was PDF.js, which is a javascript library that is built with HTML 5. It is an open source project and allows for excellent integration into the fastapi system. The original plan for a similar projects section at the bottom of the web page when viewing a document was replanned for future development. The page can be seen in Figure 7.

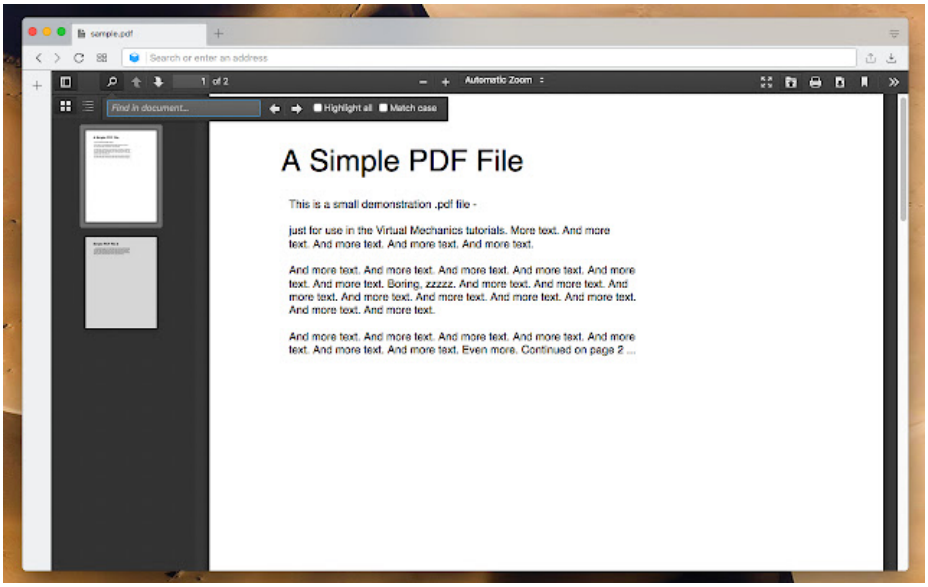

Figure 7: PDF Viewer Page of Interface.

The final design of the web interface was agreed upon by the client, Dr. Vance, and the team, Kaleb and Martin. Multiple steps were taken to plan for future development of the interface as all code will be uploaded to a GitHub repository. Dr. Vance has access to the repository and has full authority to make any changes.

4 Results and Discussion

This section summarizes the results from the user and function tests. These test plans were planned in December 2022. The Appendix contains detailed information on each individual test completed during the course of the project's development. Below, the results are discussed for each test.

4.1 User Trial Test

Initial test: January 2023.

Final test: April 2023.

Results: The user trial test provided excellent feedback that was used to redesign the interface for and improved user experience. Dr. Gallagher, Dr. Brewer, and Dr.Hill were kind enough to provide feedback on their experience with the interface. They were instructed to give feedback as if they were to be using the interface on a day-to-day basis. Constructive feedback was received including more modern design, simplified filter system, and less overwhelming results. The test was an overall success.

4.2 Browse Function Test

Initial test: March 2023.

Final test: April 2023.

Results: The browse function test confirmed an established connection between the back-end and front-end design on the browse page of the interface. The browse function correctly pulled data from the database and displayed the related projects to the user. This test was a success as seen in Figure 8.

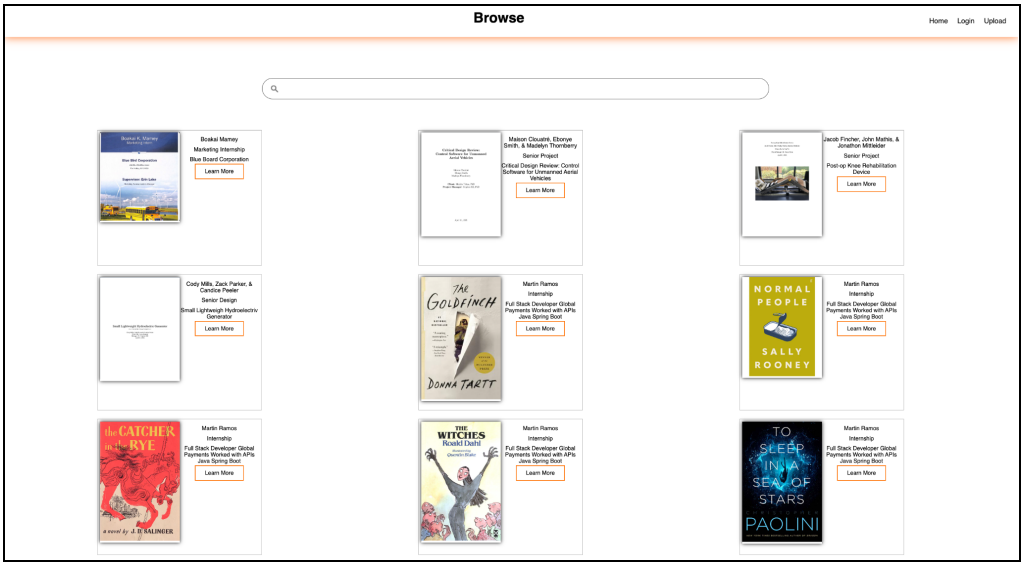

Figure 8: Results from Browse Function Test.

4.3 Upload Function Test

Initial test: March 2023.

Final test: April 2023.

Results: The upload function test made sure that when the user inputs project information into the upload form and attaches a pdf document, the database receives the correct information. The database was able to receive the information imputed by the user upon project upload as seen in Figures 9 and 10. This test was a success.

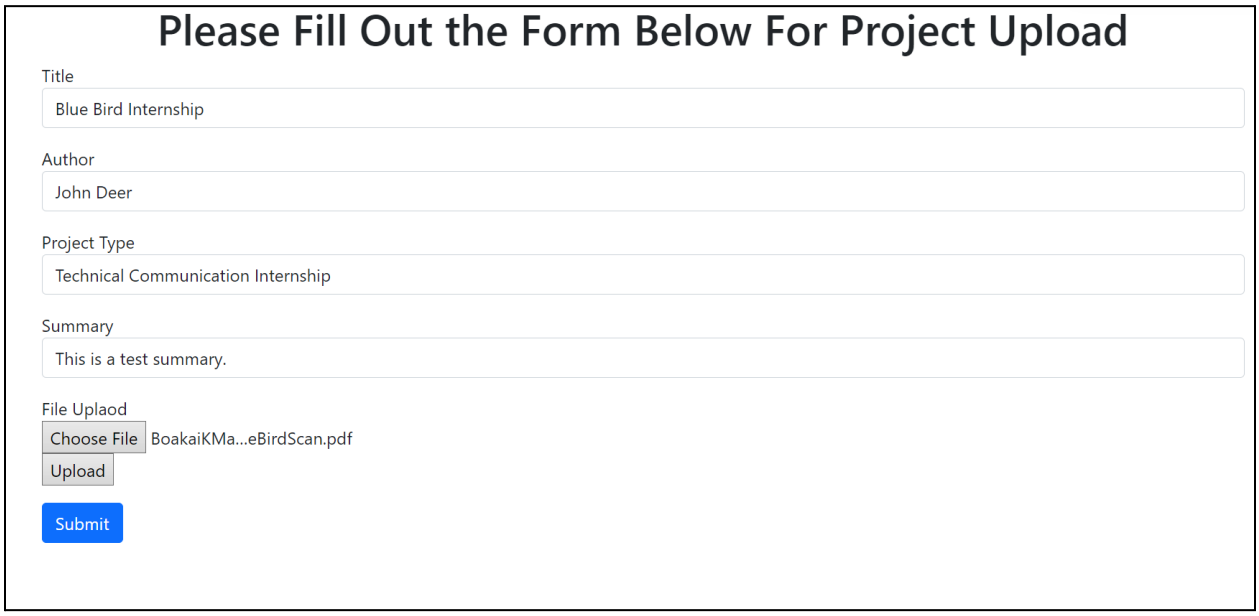

Figure 9: Information Inputted by User.

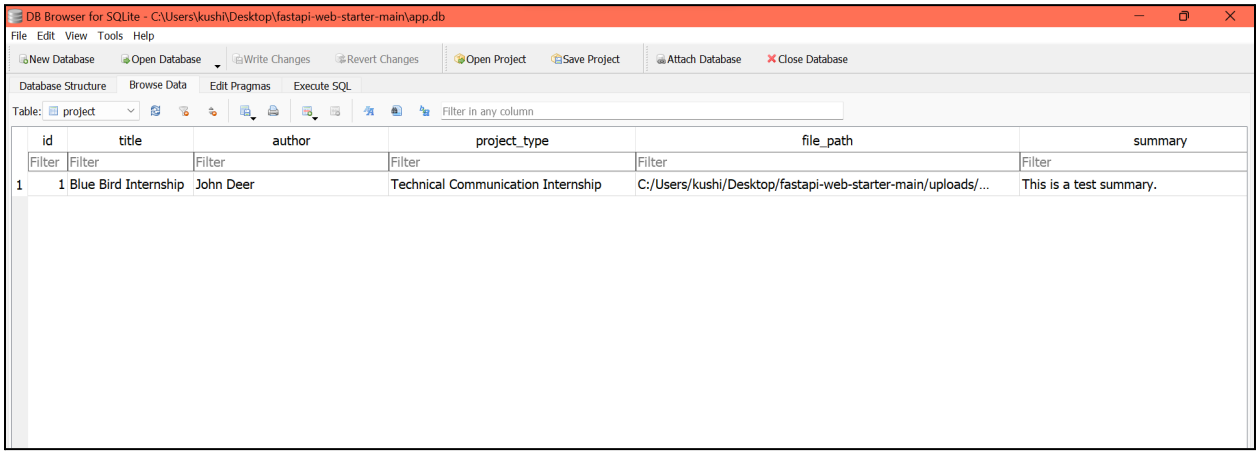

Figure 10: Database Received the Information.

4.4 PDF Viewer Test

Initial test: March 2023.

Final test: April 2023.

Results: The PDF viewer test was designed to check if the PDF viewer was able to pull the PDF document file path from the database and display it to the user. The viewer used during the testing and picked for the final design was PDF.js. It was able to properly display the document chosen by the user. This test was a success. The results of the viewer can be seen in Figure 11.

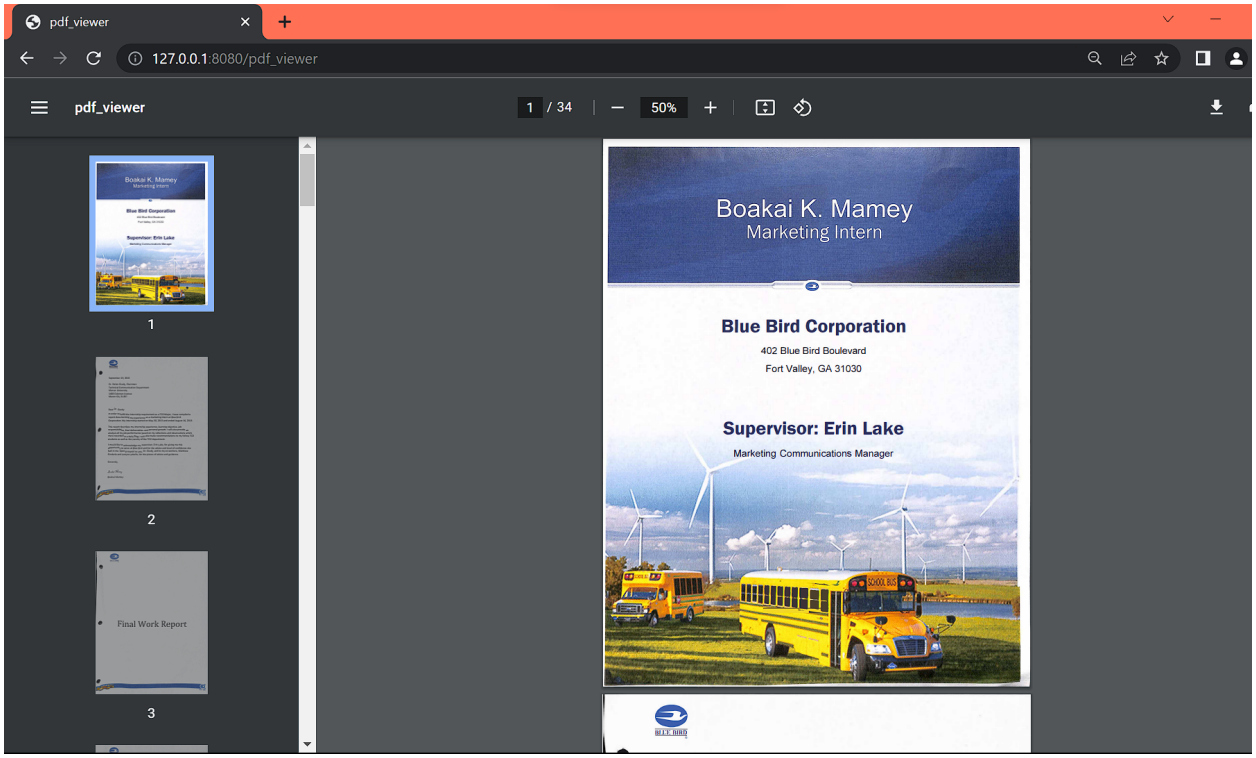

Figure 11: PDF Viewer Results.

4.5 Search Function Test

Initial test: March 2023.

Final test: April 2023.

Results: The search function test was designed to confirm the correct results would be displayed for the corresponding keywords or project information. The user inputted a requested project type and received results of the searched project type. The test was a success.

4.6 Filter Function Test

Initial test: March 2023.

Final test: April 2023.

Results: The filter function test provided feedback on if the results corresponding to the filters used would be the only results to be displayed on the results page. The user selected different filters, consisting of different combinations, and the correct results were displayed. This test was a success.

5 Summary and Conclusion

Mercer University's School of Engineering receives more project submissions every year that entails multiple project documents. These documents can be difficult for faculty and students to pull from the current physical/digital archive. Dr. Bremen Vance, our project client, has requested an online web interface to archive the project documentation in an organized catalog that students and faculty can use for research or in other educational situations. The goal of the project is to develop a platform that allows for a user to search, browse, and view project documents to adhere to the user's educational needs. Prototypes were designed and a final product was developed via Python and FastAPI coding. The goal of the project was met through the deliverables developed by the team. The code for the interface was uploaded to GitHub for future development. This procedure allows for developers in the future to be able to download the previous designed interface and continue to build upon. Recommendations for future features within the interface would be login credentials to secure the project documents from unauthorized users, an advanced search function to further pinpoint user desires, voice reading of project documents, and a feature to automatically fill out project details from an analysis upon upload.

6 References

Standards. W3C. (2021). Retrieved April 10, 2023, from https://www.w3.org/standards/

Appendix A Test Plans

A.1 User Trial Test

Type of test: Human Acceptance Test

Test Objective: The objective of this test is to determine the user experience and determine if there are any improvements to be made or accessibility needs that need to be met before the interface is ready for public use.

Equipment Needed: The equipment needed is a server(docker) setup and a computer to test the web interface.

Location: Mercer University Technical Communication Room

Date(s)/Total Time: Will begin at the end of January.

Personnel: The personnel to test the interface will be students and faculty using the library.

Criteria For Success: The criteria for success is that the interface provides a smooth and easily navigable experience for the user while trying to retrieve data from the projects they are looking for. The interface also needs to adhere to the user's needs as it meets their accessibility needs.

Procedure: The procedure will be performed by the tester, who has been given an interactive prototype of the interface to interact with. The user will use the interface in whichever means they want and write down any concerns or remarks they have while using the interface. This can be based on the function they are testing or just the design layout of each page. Then the information gathered from this test will be used to determine improvements that need to be made.

A.2 Browse Function Test

Type of test: Performance Test

Test Objective: The objective of this test is to determine the functionality of the browse function and if it is communicating with the metadata correctly to display the right projects within the right categories.

Equipment Needed: The equipment needed is a server(docker) setup and a computer to test the web interface.

Location: Mercer University Technical Communication Room

Date(s)/Total Time: Will be performed in February.

Personnel: The personnel to test the interface will be Kaleb Kushinka, Martin Ramos, and Dr. Bremen Vance.

Criteria For Success: The criteria for success is that the browse function correctly displays the right projects in the right categories based on the metadata entered for the specific project.

Procedure: The procedure will be administered in the format of writing down the specific categories the specific project should be listed under. The group will then use the browse page and check the written-down categories for the specific project.

A.3 Upload Function Test

Type of test: Performance Test

Test Objective: The objective of this test is to determine the functionality of the Upload function and if it is communicating with the metadata correctly to display the right projects within the right categories.

Equipment Needed: The equipment needed is a server(docker) setup and a computer to test the web interface.

Location: Mercer University Technical Communication Room

Date(s)/Total Time: Will be performed at the beginning of April.

Personnel: The personnel to test the interface will be Kaleb Kushinka, Martin Ramos, and Dr. Bremen Vance.

Criteria For Success: The criteria for success is that the Upload function most correctly collects data from the pdf file upload and fills out the metadata fields below the upload button but also allows the user to manually override them if they see anything that is not filled out correctly.

Procedure: The procedure will be performed by the tester going to the upload page and uploading a pdf project to the database. The user will then check if the upload is a success by the interface indicating that the upload was complete. The user will then check the metadata filled out by the interface is mostly correct. The interface will most likely not be able to analyze the exact description of the project by the analysis but will be very close. The user will then fill in any necessary data for the project and hit submit. The final check will be to go into the database and confirm the metadata has been entered and registered by the interface.

A.4 PDF Viewer Test

Type of test: Performance Test

Test Objective: The objective of this test is to determine the functionality of the PDF Viewer function and if it displays the project correctly. It also is to check if the accessibility features work, such as keyboard functionality.

Equipment Needed: The equipment needed is a server(docker) setup and a computer to test the web interface.

Location: Mercer University Technical Communication Room

Date(s)/Total Time: Will be performed in February.

Personnel: The personnel to test the interface will be Kaleb Kushinka, Martin Ramos, and Dr. Bremen Vance.

Criteria For Success: The criteria for success is that the PDF Viewer function most correctly displays the project for the user to read and displays similar projects at the end of the page. It also needs to adhere to the accessibility of users by allowing keyboard functionality.

Procedure: The procedure will be performed by the tester clicking on a project to view. The user will test to see if the PDF viewer displays the project correctly and allows for the user to zoom in and flip through the pages. The user will then test the keyboard functionality and see if they are able to flip between the pages by using the arrow keys on the keyboard. The user will then look to the bottom of the screen to see if there are similar projects to also view with a small little summary for the user to get a feel for if they want to read that project or not depending on the purpose of their library use.

A.5 Search Function Test

Type of test: Performance Test

Test Objective: The objective of this test is to determine the functionality of the Search function and if it is communicating with the metadata correctly to display the right projects within the right categories.

Equipment Needed: The equipment needed is a server(docker) setup and a computer to test the web interface.

Location: Mercer University Technical Communication Room

Date(s)/Total Time: Will be performed in March.

Personnel: The personnel to test the interface will be Kaleb Kushinka, Martin Ramos, and Dr. Bremen Vance.

Criteria For Success: The criteria for success is that the Search function correctly displays the right projects in the right categories based on the metadata entered for the specific project. The results page should appear with projects pertaining to only the specific phrase or words. The search function should also predict the phrase the user is trying to search for.

Procedure: The procedure will be performed by searching for projects using specific phrases or words. The group will then make sure that each of the projects showing within the phrase or word the user is trying to search for. If the page displays any projects not pertaining to the phrase, then the function needs to be reworked.

A.6 Filter Function Test

Type of test: Performance Test

Test Objective: The objective of this test is to determine the functionality of the filter function and if it is communicating with the metadata correctly to display the right projects within the right categories.

Equipment Needed: The equipment needed is a server(docker) setup and a computer to test the web interface.

Location: Mercer University Technical Communication Room

Date(s)/Total Time: Will be performed in March.

Personnel: The personnel to test the interface will be Kaleb Kushinka, Martin Ramos, and Dr. Bremen Vance.

Criteria For Success: The criteria for success is that the filter function correctly displays the right projects in the right categories based on the metadata entered for the specific project.

Procedure: The procedure will be administered in the format of writing down the specific categories the specific project should be listed under. The group will then use the filter function to alter the browse page to show only aspects pertaining to that project to be displayed and check the written-down categories for the specific project.

Appendix B Sample Front-End Code

styleHome.css ● # styleBrowse.css ×

styleBrowse.css > section header

```
80  .results .results_box .results_card
81  {
82      width: 350px;
83      height: 290px;
84      text-align: center;
85      padding: 5px;
86      border: 1px solid #a2a2a2;
87      margin: auto 145px;
88  }
89  .results .results_box .results_card:hover
90  {
91      box-shadow: 0 0 5px rgb(255, 111, 0)
92  }
93
94  .results .results_box .results_card .results_image
95  {
96      width: 50%;
97      margin: 0 auto;
98      cursor: pointer;
99      box-shadow: 0 0 8px rgb(0,0,0);
100     overflow:hidden;
101     float: left;
102
103 }
104
105 .results .results_box .results_card .results_image img
106 {
107     width: 100%;
108     height: 100%;
109     object-fit: cover;
110     object-position: center;
111     transition:0.3s;
112 }
113 .results .results_box .results_card:hover .results_image img
114 {
115     transform: scale(1.1);
116 }
117 .results .results_box .results_card .results_tag p
118 {
119     font-family: sans-serif;
120     font-size: 12px;
121     margin: 8px 0;
122 }
123 .results .results_box .results_card .results_tag .results_btn
124 {
125     padding: 8px 20px;
126     border: 2px solid rgb(255, 111, 0);
127     text-decoration: none;
128     color: #000;
129     font-size: 12px;
130 }
131
```

styleHome.css 1

styleHome.css > ...

```css
section .main
{
    width: 580px;
    display: flex;
    flex-direction: column;
    align-items: center;
    justify-content: center;
}

section .main .title1
{
    display: flex;
}
section .main .title1 li
{
    list-style: none;
    margin: 0 5px;
    font-size: 38px;
    color: rgb(255, 111, 0);
    font-family:sans-serif ;
    font-weight: bold;
}
section .main .title2
{
    display: flex;
}

section .main .title2 li
{
    list-style: none;
    margin: 0 5px;
    font-size: 38px;
    font-family: sans-serif;
    font-weight: bold;
}
section .main .searchBox
{
    position: relative;
    width: 100%;
    margin-top: 20px;
}
section .main .searchBox .search
{
    width: 100%;
    padding: 13px;
    padding-left: 45px;
    padding-right: 60px;
    border-radius: 30px;
    border: 1px solid #ccc;
    outline: none;
    font-size: 16px;
}
```

styleBrowse.css

```css
section header
{
    position: absolute;
    top: 0;
    width: 100%;
    display: flex;
    justify-content: flex-end;
    padding: 25px;
    box-shadow: 0 0 20px rgb(255, 111, 0);
}
section header ul
{
    display: flex;
    justify-content: center;
    align-items: center;
}
section header ul li
{
    list-style: none;
    margin-left: 20px;
}
section header ul li a
{
    color: #111;
    text-decoration: none;
    font-size: 15px;
}
section header ul li a:hover
{
    color: rgb(255, 111, 0);
}

/* results */

.results{
    width: 100%;
    height: 100vh;
    margin-bottom: 35px;
    display: flex;
    flex-wrap: wrap;

}

.results h1{
    font-size: 50px;
    text-align: center;
    margin-bottom: 35px;
}
```

browse.html

```
<!DOCTYPE html>

<html lang="en">
<head>
    <meta charset="utf-8" />
    <title>Mercer Projects Library</title>
    <link rel="stylesheet" href="styleBrowse.css">
</head>
<body>
    <section>
        <header>
            <ul>
                <li><a href="#">Login</a></li>
                <li><a href="#">Upload</a></li>
            </ul>
        </header>
        <div class="results">
            <h1>Results</h1>

            <div class="results_box">

               <div class="results_card">
                    <div class="results_image">
                        <img src="images/results_1.jpeg" alt="">
                    </div>
                    <div class="results_tag">
                        <p class="author">Martin Ramos</p>
                        <p class="type">Internship</p>
                        <p class="summary">Full Stack Developer Global Payments</p>
                        <a href="#" class="results_btn">Learn More</a>
                    </div>
               </div>

            <div class="results_card">
                <div class="results_image">
                    <img src="images/results_2.jpeg" alt="">
                </div>
                <div class="results_tag">
                        <p class="author">Martin Ramos</p>
                        <p class="type">Internship</p>
                 <p class=" summary">Full Stack Developer Global Payments Worked with APIs Java Spring Boot</p>
                    <div class="results_icon">

```

home.html

```html
<!DOCTYPE html>

<html lang="en">
<head>
    <meta charset="utf-8" />
    <title>Mercer Projects Library</title>
    <link rel="stylesheet" href="styleHome.css">
</head>
<body>
    <section>
        <header>
            <ul>
                <li><a href="#">Login</a></li>
                <li><a href="#">Browse</a></li>
                <li><a href="#">Upload</a></li>
            </ul>
        </header>
        <div class="main">
            <div class="logo">
                <img src="logo.png">
            </div>
            <ul class="title1">
                <li>Mercer University</li>
            </ul>
            <ul class="title2">
                <li>Online Project Library</li>
            </ul>

            <div class="searchBox">
                <input type="text" class="search">
                <div class="icons">
                    <div><img src="Search.png"></div>
                </div>
            </div>
            <div class="buttons">
                <button> Not Sure? Click Here to Browse </button>
            </div>
            <ul class="description">
                <li>Welcome to the Online Project Library.<br> Here you can find a repository of projects<br>documentation over the years.</li>
            </ul>
        </div>
        <div class="footer">
            <div class="row">
            </div>
        </div>
    </section>
</body>
</html>
```

Appendix C Sample Back-End Code

```python
from sqlalchemy import create_engine
from sqlalchemy.ext.declarative import declarative_base
from sqlalchemy.orm import sessionmaker

SQLALCHEMY_DATABASE_URL = "sqlite:///app.db"
# SQLALCHEMY_DATABASE_URL = "postgresql://user:password@postgresserver/db"

engine = create_engine(
    SQLALCHEMY_DATABASE_URL, connect_args={"check_same_thread": False}
)
SessionLocal = sessionmaker(autocommit=False, autoflush=False, bind=engine)

Base = declarative_base()
```

```python
from sqlalchemy import Boolean, Column, ForeignKey, Integer, String
from sqlalchemy.orm import relationship

from .database import Base

class Project(Base):
    __tablename__ = "project"

    id = Column(Integer, primary_key=True, index=True)
    title = Column(String, index=True)
    author = Column(String, index=True)
    project_type = Column(String, index=True)
    summary = Column(String, index=True)
    file_path = Column(String(255), index=True)



def __repr__(self):
        return '<Project %r>' % (self.id)
```

```python
from typing import List, Union

from pydantic import BaseModel

class ProjectBase(BaseModel):
    title: str
    author: str
    project_type: str
    summary: str
    file_path: str

class ProjectCreate(ProjectBase):
    pass

class Project(ProjectBase):
    id: int
    title: str
    author: str
    project_type: str
    summary: str
    file_path: str

    class Config:
        orm_mode = True

```

```python
from typing import List

from fastapi import FastAPI, Form, status, Request, Depends, HTTPException, File, UploadFile, Response
from fastapi.responses import HTMLResponse, StreamingResponse
from fastapi.staticfiles import StaticFiles
from fastapi.responses import RedirectResponse
from fastapi.templating import Jinja2Templates
from pathlib import Path

from sqlalchemy import create_engine, Column, Integer, String
from sqlalchemy.ext.declarative import declarative_base
from sqlalchemy.orm import Session
from sqlalchemy.orm import sessionmaker
from .import crud, models, schemas
from .database import SessionLocal, engine

from .library.helpers import *
from app.routers import twoforms, unsplash, accordion

models.Base.metadata.create_all(bind=engine)

app = FastAPI()
Base = declarative_base()


#Dependency
def get_db():
    db = SessionLocal()
    try:
        yield db
    finally:
        db.close()

templates = Jinja2Templates(directory="templates")
```

```python
app.mount("/static", StaticFiles(directory="static"), name="static")

app.include_router(unsplash.router)
app.include_router(twoforms.router)
app.include_router(accordion.router)


@app.get("/pdf_viewer")
async def get_pdf():
    with open("uploads/BoakaiKMameyBlueBirdScan.pdf", "rb") as pdf_file:
        pdf = pdf_file.read()
    return Response(content=pdf, media_type="application/pdf")
```

```python
@app.get("/")
async def root(request: Request):
    html_content = """
    <!DOCTYPE html>
    <html>
    <head>
        <title>PDF.js Example</title>
        <script src="https://cdnjs.cloudflare.com/ajax/libs/pdf.js/2.11.338/pdf.min.js"></scr
    </head>
    <body>
        <div>
            <canvas id="pdf-canvas"></canvas>
        </div>
        <script>
            pdfjsLib.getDocument("/pdf_viewer").promise.then(function(pdf) {
                pdf.getPage(1).then(function(page) {
                    var canvas = document.getElementById("pdf-canvas");
                    var context = canvas.getContext("2d");
                    var viewport = page.getViewport({scale: 1});
                    canvas.height = viewport.height;
                    canvas.width = viewport.width;
                    page.render({canvasContext: context, viewport: viewport});
                });
            });
        </script>
    </body>
    </html>
    """
    return HTMLResponse(content=html_content, status_code=200)
```

```python
@app.get("/index")
async def home(request: Request, db: Session = Depends(get_db)):
    project = db.query(models.Project).order_by(models.Project.id.desc())
    return templates.TemplateResponse("index.html", {"request": request, "project": project})

@app.post("/add_project")
async def add(request: Request, title: str = Form(...), author: str = Form(...), project_type: str = Form(...),summary: str = Form(...), file: UploadFile = File(...), db: Session = D
    contents = await file.read()
    with open(f"C:/Users/kushi/Desktop/fastapi-web-starter-main/uploads/{file.filename}","wb") as f:
        f.write(contents)

    new_file = models.Project(title=title, author=author, project_type=project_type, summary = summary, file_path=f"C:/Users/kushi/Desktop/fastapi-web-starter-main/uploads/{file.file

    db.add(new_file)
    db.commit()
    db.refresh(new_file)
    return RedirectResponse(url=app.url_path_for("home"), status_code=status.HTTP_303_SEE_OTHER)

@app.get("/basee")
async def addnew(request: Request):
    return templates.TemplateResponse("basee.html", {"request": request})


@app.get("/delete/{project_id}")
async def delete(request: Request, project_id: int, db: Session = Depends(get_db)):
    project = db.query(models.Project).filter(models.Project.id == project_id).first()
    db.delete(project)
    db.commit()
    return RedirectResponse(url=app.url_path_for("home"), status_code=status.HTTP_303_SEE_OTHER)
```

Appendix D Metadata Excel Sheet

Title	Author	Job Type	Year Complete	Employer	Supervisior	Includes Digital Mec
TCO Internship	Andrea LaPlume	Study Abroad Intern	2003	Bond University	Brad Dorahy	No
BrightWave Marketing Digital Media Marketing Internship	Zachary Norman	Digital Media Marketing	2011	BrightWave Marketing	Ryan Tuttle	No
E-Commerece & Web Design	Roxanne Perkins	Web Designer	1999	Camellia & Main	Jim Swift	Yes
Blue Bird Coperation	Boakai K. Mamey	Marketing Intern	2013	Blue Bird Corporation	Erin Lake	No
Safety Internship	D'Arlous Madden	Safety Intern	2013	Blue Bird Corporation	Robert Watts	No
Blue Bird Marketing Internship	Patrick Heise	Marketing Intership	2015	Blue Bird Corporation	Justyne Lobell	No
Blue Bird	Robert James Purse	Marketing Intern	2014	Blue Bird Corporation	Matthew Rinde	No
Technical Communiation Internship	Erik Lindborg	Technical Publications Intern	2003	Alpha Sports Motors & Asian Ventures, Inc	David Turner	Yes
Blue Bird Coperation	Dustin Franklin	Marketing Intern	2011	Blue Bird Corporation	Erin Lake	No
Manuals and Materuaks Internship	Nakita K. James	Manuals & Materials Internship	2013	National Alpha Lambda Delta Honor Socie	Lee Greenway	No
All Access Computers	Kevin Woodall	Internet Programming Intern	2010	All Access Computers	Bennie Colema	No
Countrywide Promotions	Ken Demuyakor	Website Administrator	2007	Countrywide Promotions	Clark Haddock	No
AT&T FLITE Instructional Design Intern	Alejandro Granda M	Instructional Design Intern	2015	AT&T FLITE	James Brunet	No
Instructional Design Internship Report	Jake Missall	Instructional Design Intern	2016	Mercer Engineering Research Center	N/A	No
Final Work Report	Al Skelton	Web Developer	1999	ARINC, Inc	Del Olds	No
Final Work Report	Ed Johnson	technical Writer Intern	2003	Computer Process Controls	Matt Lauck	Yes
Internship Report	Rick Tresco	DA Multimedia	1996	Documentation Associates	Mark Hagan	No
Technical Writer	Pascal Occean	technical Writer Intern	1998	ABL Canada	John Berendse	No
Cingular Wireless	Rachel Astorino	IT Analyst	2006	Cigular Wireless	Colleen Jones	No
ChoicePoint	Kati Watson	Technical Communciation Intern	2003	ChoicePoint	N/A	No

Deliverables	Includes Daily Log	Includes Final Report	Includes Final Presentatio	Department Reccomendations	Includes Sample Deliverable
Study Broad Guide, Bibliography, Newsletter	Yes	Yes	Yes	Second Multimedia Course, Send more students Al	Yes
Handbook Creation, Email Portfolio, Quick start Guides	Yes	Yes	No	Include Computer Eningeering Principles, Underste	Yes
Web Design, Post Card Design	Yes	Yes	No	Include a list of possible jobs	Yes
Fliers, Web Developments, Video Production, Image Edits	Yes	Yes	No	Include Web Development course, Include more we	Yes
Training Development, Instructional Design, Mapping, Report Cre	Yes	Yes	No	Form Interddepartmental Connection with Commun	Yes
Design, Web Design, Marketing Materials	Yes	Yes	No	Learn Content Managment Systems	Yes
Brochure, Ad Design, Fliers	Yes	Yes	No	Indepth Adobe training	Yes
Manuals,Documents, IT Taks, Web Design	Yes	Yes	Yes	Adobe Training, Technical Editing requirment, Fran	Yes
Ads, Logos, fliers, Web Banners, Brochures, Web Design	Yes	Yes	No	Adobe Training	Yes
User Manual, Brochure, Postcard, Fundrasing letters	Yes	Yes	No	Look for internships earlier, Collaborate with carees	Yes
Web Design, Ebook Creation	Yes	Yes	No	Adobe Training	Yes
Web Design	Yes	Yes	No	Include Computer Engineering Principles	Yes
Technical Editing, Web Design, Instructional Design	Yes	Yes	No	Include Computer Engineering Principles/ Coding i	No
Instructional Design	Yes	Yes	No	Continue Hands on training	Yes
Web Design	Yes	Yes	No	Implimentation of a mentor program, More emphasi	Yes
Instructional Design	Yes	Yes	No	Documentation Creation Process, Additional trainy	Yes
Template Creation, Document Design, Information managment	No	Yes	No	None	Yes
Edits and Updates, Document Design, Translation	Yes	Yes	No	Preperation for writing manuals	Yes
Copyediting, Guide Creation	Yes	Yes	No	Flash Training, HTML coding skills	Yes
Manuals, Documentation	Yes	Yes	Yes	Require A business class, Establish relationships w	No