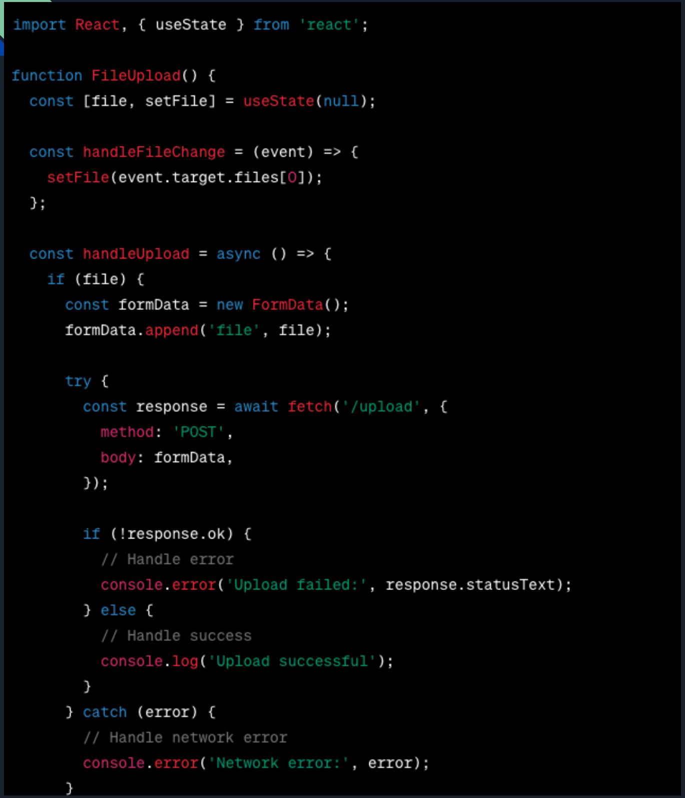
Web application

Justin Jung & Leo Kuo

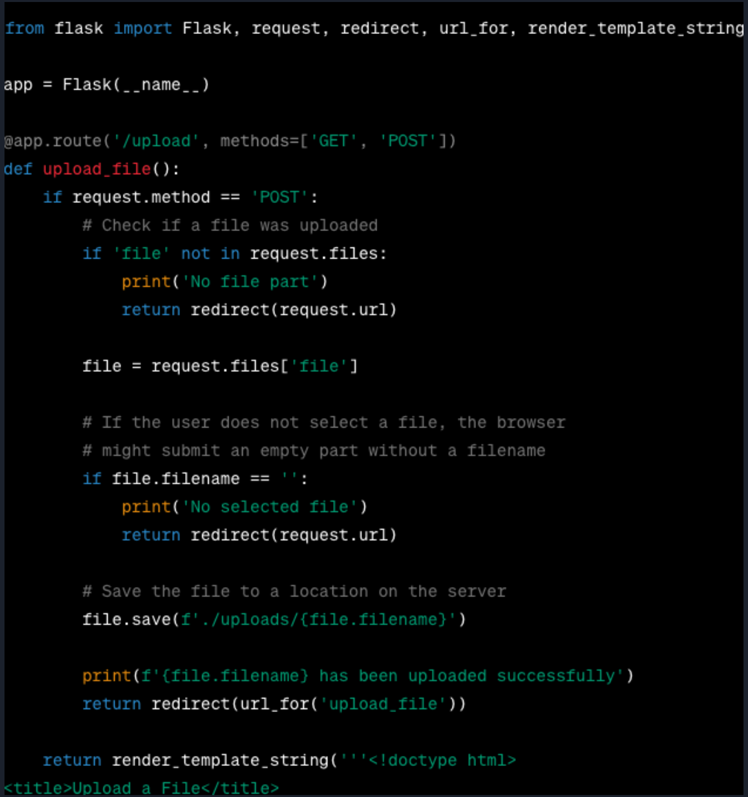
What I have done for the project

- Researched for how to upload DICOM images from local device to the backend(pre-built component)
- In the backend we will be using framework such as Flask or Django(Most likely Flask) to create RESTful API.
- Read DICOM images through Library like PyDicom
- Pros/cons encode/decode DICOM images in base64(Flask reads binary data. Base64 increase size by 33%)
- How to create progress bar for progress feedback(frontend)

Upload and Read DICOM images

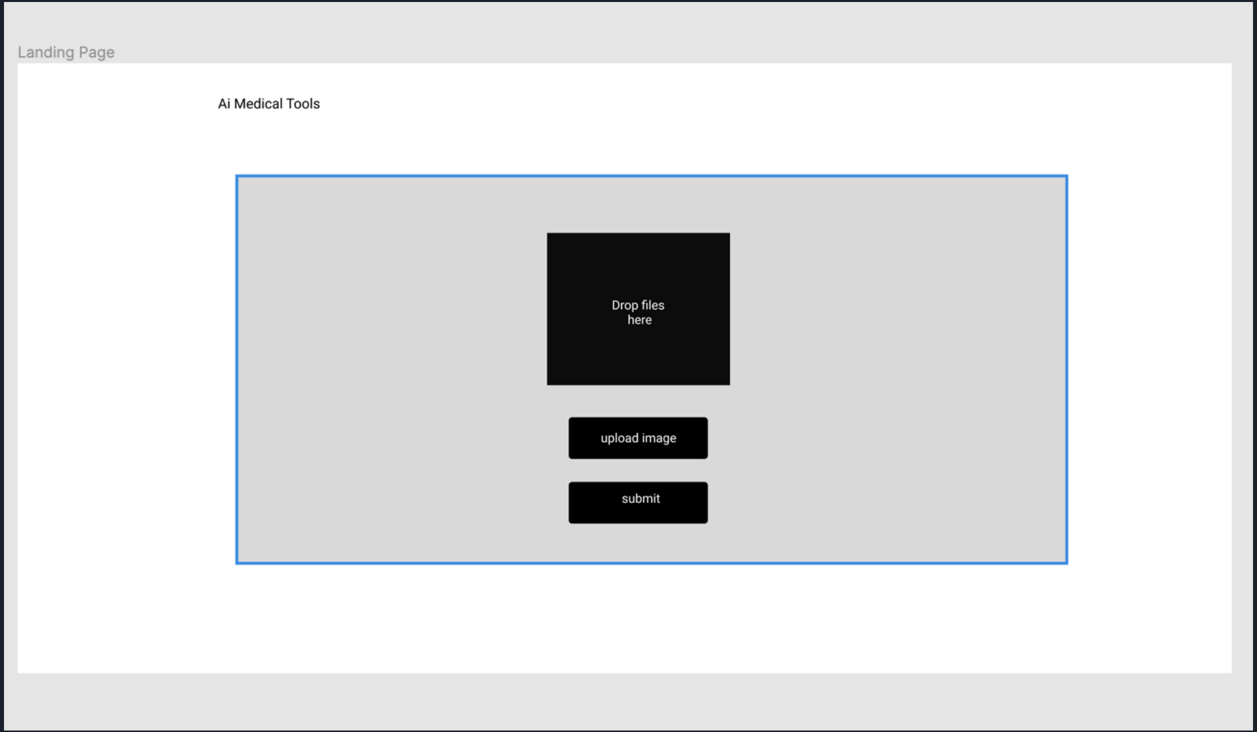


```javascript
import React, { useState } from 'react';

function FileUpload() {
  const [file, setFile] = useState(null);

  const handleFileChange = (event) => {
    setFile(event.target.files[0]);
  };

  const handleUpload = async () => {
    if (file) {
      const formData = new FormData();
      formData.append('file', file);

      try {
        const response = await fetch('/upload', {
          method: 'POST',
          body: formData,
        });

        if (!response.ok) {
          // Handle error
          console.error('Upload failed:', response.statusText);
        } else {
          // Handle success
          console.log('Upload successful');
        }
      } catch (error) {
        // Handle network error
        console.error('Network error:', error);
      }
```

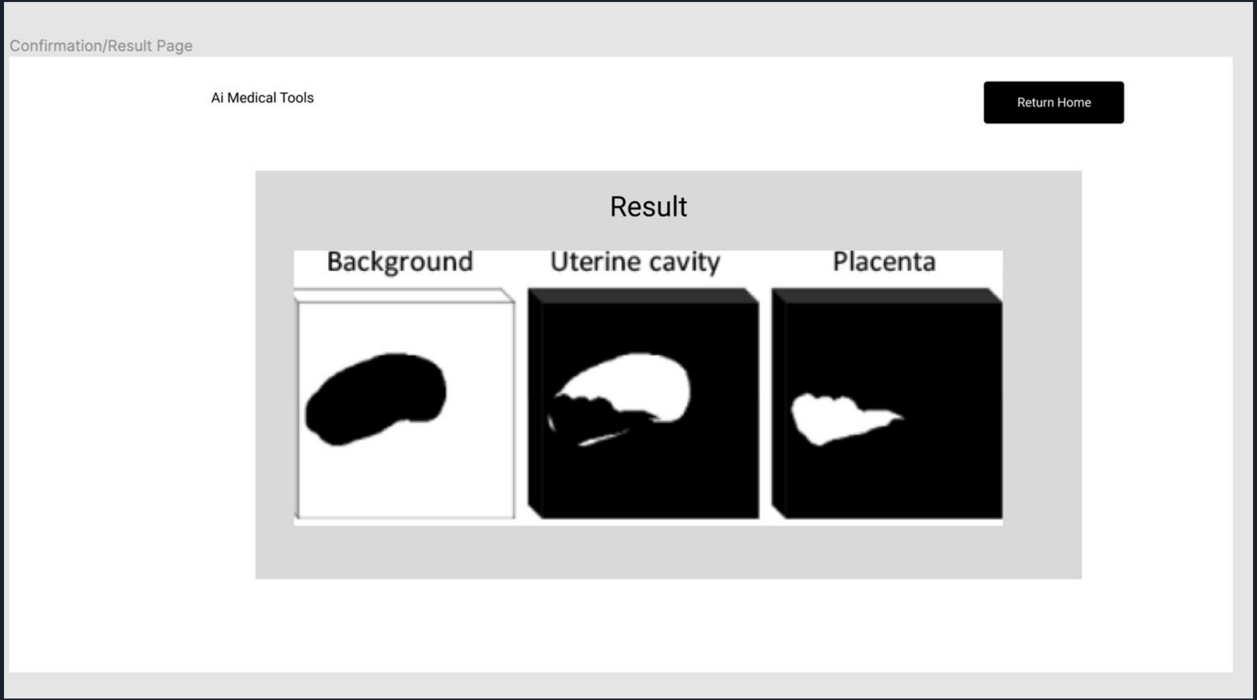


```python
from flask import Flask, request, redirect, url_for, render_template_string

app = Flask(__name__)

@app.route('/upload', methods=['GET', 'POST'])
def upload_file():
    if request.method == 'POST':
        # Check if a file was uploaded
        if 'file' not in request.files:
            print('No file part')
            return redirect(request.url)

        file = request.files['file']

        # If the user does not select a file, the browser
        # might submit an empty part without a filename
        if file.filename == '':
            print('No selected file')
            return redirect(request.url)

        # Save the file to a location on the server
        file.save(f'./uploads/{file.filename}')

        print(f'{file.filename} has been uploaded successfully')
        return redirect(url_for('upload_file'))

    return render_template_string('''<!doctype html>
<title>Upload a File</title>
```

WireFrame- landing page



Confirmation page



Future Tasks

- Create landing page and confirmation page
- Upload function
- Submit function
- Router of the website
- Create backend APIs
- Implement APIs and connect to Next.js backend

Docker FIle

Dockerfile to ensure that the code works on majority of the platforms

Testing

We will log the data of the images and server response throughout the development stage.

After the prototype is finished, we will show to Dr. Fei to get feedbacks and make adjustments.

Timeline

Week	Action
Week1(10/10-10/17)	Setup development environment. Finalize project architecture. Version control practice. Establish coding guidelines.
Phase 2 (10/18-10/31)	Develop the frontend: Landing and confirmation page. Backend: Create APIs
Phase 3(11/1-11/14)	Continue with backend development. Test the APIs. Frontend: finish image upload part and test image
Phase 4(11/15-11/24)	Finish up frontend part: display image and test the data flow within the system. Identify issues and fix them.
Phase 5(11/25-12/1)	Deployment and documentation.
Phase 6(12/9)	presentation

END
Thank you