LABORATORY REPORT

# Application Development Lab
# (CS33002)

## B.Tech Program in ECSc

Submitted By

**Name:-** Bhairav Ganguly

**Roll No:** 2230246


**Kalinga Institute of Industrial Technology**
**(Deemed to be University)**
**Bhubaneswar, India**

Spring 2024-2025

# Table of Content

| | |
|---|---|
| **Experiment Number** | 9 |
| **Experiment Title** | Frontend report of OCR PDF & Chat |
| **Date of Experiment** | 02/04/25 |
| **Date of Submission** | 05/04/25 |

**1. Objective:-** The objective of this lab experiment is to create a Python program that converts handwritten notes and digital text from PDF files into digital text, enhancing accessibility. Additionally, it allows users to interact with the PDF content through natural language processing for efficient information retrieval and user engagement.

**2. Procedure:-** The project involved using Python as the core programming language to integrate Google Vision API for optical character recognition (OCR) to extract text from handwritten notes and digital text PDFs. For the interactive chat functionality with the PDF content, the Groq API Llama 3-8b model was employed to implement natural language processing. The frontend of the application was developed using Vanilla JavaScript, HTML, and CSS, further styled with Tailwind CSS to ensure a user-friendly and responsive interface.

## 3. Code:-

Base.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>PDF OCR Chat</title>
   <link href="https://cdn.jsdelivr.net/npm/tailwindcss@2.2.19/dist/tailwind.min.css" rel="stylesheet">
   <link rel="stylesheet" href="{{ url_for('static', filename='css/style.css') }}">
```

```
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/socket.io/4.0.1/socket.io.js"><
/script>
    <style>
        .chat-message div:last-child
            { white-space: pre-line;
        }
        .chat-message ul {
            list-style-type: disc;
            padding-left: 20px;
            margin-top: 5px;
            margin-bottom: 5px;
        }
        .chat-message li
            { margin-bottom:
            8px;
        }
    </style>
</head>
<body class="bg-gray-900 text-gray-100 min-h-screen">
    <div class="container mx-auto px-4 py-8">
        {% block content %}{% endblock %}
    </div>
    <script src="{{ url_for('static', filename='js/script.js') }}"></script>
</body>
</html>
```

Chat.html:

```
{% extends "base.html" %}

{% block content %}
<div class="max-w-4xl mx-auto">
    <h1 class="text-3xl font-bold mb-6">Chat with your PDF</h1>

    <div class="bg-gray-800 rounded-lg shadow-lg p-4 h-[60vh] overflow-
y-auto mb-4" id="chat-container">
        <div class="chat-message bg-gray-700 p-3 rounded-lg mb-3">
            <div class="font-bold text-blue-400">AI Assistant</div>
            <div>I've processed your PDF. You can now ask me questions
about its content.</div>
        </div>
    </div>
```

```
    <div class="flex space-x-2">
      <input type="text" id="user-input" placeholder="Ask a question about the PDF..."
             class="flex-1 bg-gray-700 border border-gray-600 rounded-lg px-4 py-2 focus:outline-none focus:ring-2 focus:ring-blue-500">
      <button id="send-button" class="bg-blue-600 hover:bg-blue-700 text-white font-bold py-2 px-6 rounded-lg">
        Send
      </button>
    </div>

    <div class="mt-4 text-sm text-gray-400">
      <p>Powered by Google Cloud Vision (OCR) and Groq (Llama3-8b-8192)</p>
    </div>
  </div>
  {% endblock %}
```

Upload.html:

```
{% extends "base.html" %}

{% block content %}
<div class="max-w-2xl mx-auto">
  <h1 class="text-3xl font-bold mb-6 text-center">PDF OCR & Chat</h1>
  <div class="bg-gray-800 rounded-lg shadow-lg p-6">
    <form method="POST" enctype="multipart/form-data" class="space-y-4">
      <div>
        <label class="block text-sm font-medium mb-2">Upload PDF File</label>
        <input type="file" name="file" accept=".pdf" class="block w-full text-sm text-gray-400
          file:mr-4 file:py-2 file:px-4
          file:rounded-md file:border-0
          file:text-sm file:font-semibold
          file:bg-blue-500 file:text-white
          hover:file:bg-blue-600
          cursor-pointer">
      </div>
      <button type="submit" class="w-full bg-blue-600 hover:bg-blue-700 text-white font-bold py-2 px-4 rounded">
```

```
            Process PDF
        </button>
    </form>
</div>
<div class="mt-8 text-gray-400 text-sm">
    <p>This application will:</p>
    <ol class="list-decimal pl-5 mt-2 space-y-1">
        <li>Extract text from your PDF (or perform OCR if needed)</li>
        <li>Allow you to chat with the document using AI</li>
    </ol>
</div>
</div>
{% endblock %}
```

JS Code:

```
document.addEventListener('DOMContentLoaded', function()
    { const socket = io();
    const chatContainer = document.getElementById('chat-container');
    const userInput = document.getElementById('user-input');
    const sendButton = document.getElementById('send-button');

    function addMessage(role, message, isHTML = false)
        { const messageDiv = document.createElement('div');
        messageDiv.className = `chat-message bg-gray-700 p-3 rounded-lg
mb-3`;

        const roleDiv = document.createElement('div');
        roleDiv.className = role === 'user' ? 'font-bold text-green-400' :
'font-bold text-blue-400';
        roleDiv.textContent = role === 'user' ? 'You' : 'AI Assistant';

        const contentDiv = document.createElement('div');
        if (isHTML) {
            contentDiv.innerHTML = message;
        } else {
            contentDiv.textContent = message;
        }

        messageDiv.appendChild(roleDiv);
        messageDiv.appendChild(contentDiv);
        chatContainer.appendChild(messageDiv);
        chatContainer.scrollTop = chatContainer.scrollHeight;
```

```
    }

    function sendMessage() {
        const message = userInput.value.trim();
        if (message) {
            addMessage('user', message);
            userInput.value = '';

            // Show typing indicator
            const typingDiv = document.createElement('div');
            typingDiv.id = 'typing-indicator';
            typingDiv.className = 'chat-message bg-gray-700 p-3 rounded-lg
mb-3';
            typingDiv.innerHTML = '<div class="font-bold text-blue-400">AI
Assistant</div><div class="flex space-x-1"><div class="w-2 h-2 bg-gray-
400 rounded-full animate-bounce"></div><div class="w-2 h-2 bg-gray-
400 rounded-full animate-bounce" style="animation-delay:
0.2s"></div><div class="w-2 h-2 bg-gray-400 rounded-full animate-
bounce" style="animation-delay: 0.4s"></div></div>';
            chatContainer.appendChild(typingDiv);
            chatContainer.scrollTop = chatContainer.scrollHeight;

            socket.emit('send_message', { message: message });
        }
    }

    sendButton.addEventListener('click', sendMessage);
    userInput.addEventListener('keypress', function(e) {
        if (e.key === 'Enter')
            { sendMessage();
        }
    });

    socket.on('receive_message', function(data) {
        const typingIndicator = document.getElementById('typing-
indicator');
        if (typingIndicator)
            { typingIndicator.remove();
        }
        addMessage('assistant', data.message, true);
    });
});
```

CSS code:

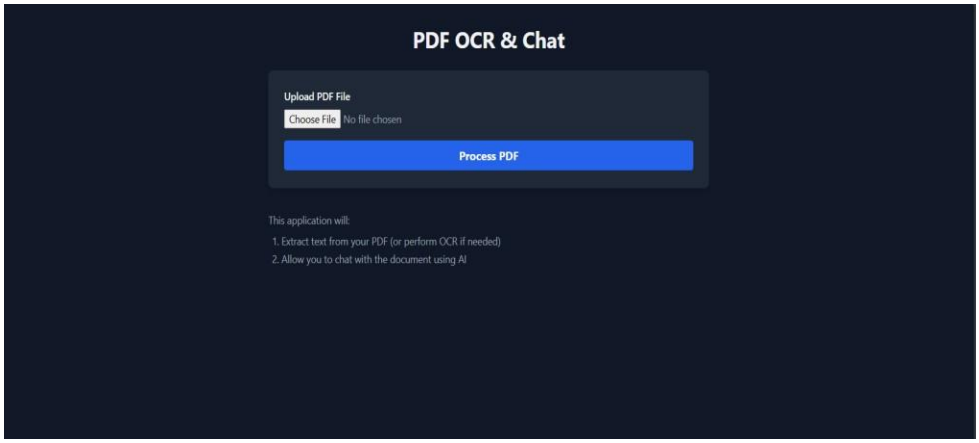
```
/* Custom scrollbar */
::-webkit-scrollbar
    { width: 8px;
}

::-webkit-scrollbar-track
    { background: #374151;
}

::-webkit-scrollbar-thumb
    { background: #4B5563;
    border-radius: 4px;
}

::-webkit-scrollbar-thumb:hover
    { background: #6B7280;
}

/* Chat container styling */
#chat-container {
    scrollbar-width: thin;
    scrollbar-color: #4B5563 #374151;
}

.chat-message {
    animation: fadeIn 0.3s ease-in-out;
}

/* Add this to your CSS */
.message {
    margin-bottom: 15px;
    line-height: 1.5;
}
.message ul
    { margin-top:
    10px;
    margin-bottom: 15px;
    padding-left: 20px;
}
.message li {
    margin-bottom: 8px;
}
```

```
@keyframes fadeIn {
   from { opacity: 0; transform: translateY(10px); }
   to { opacity: 1; transform: translateY(0); }
}
```

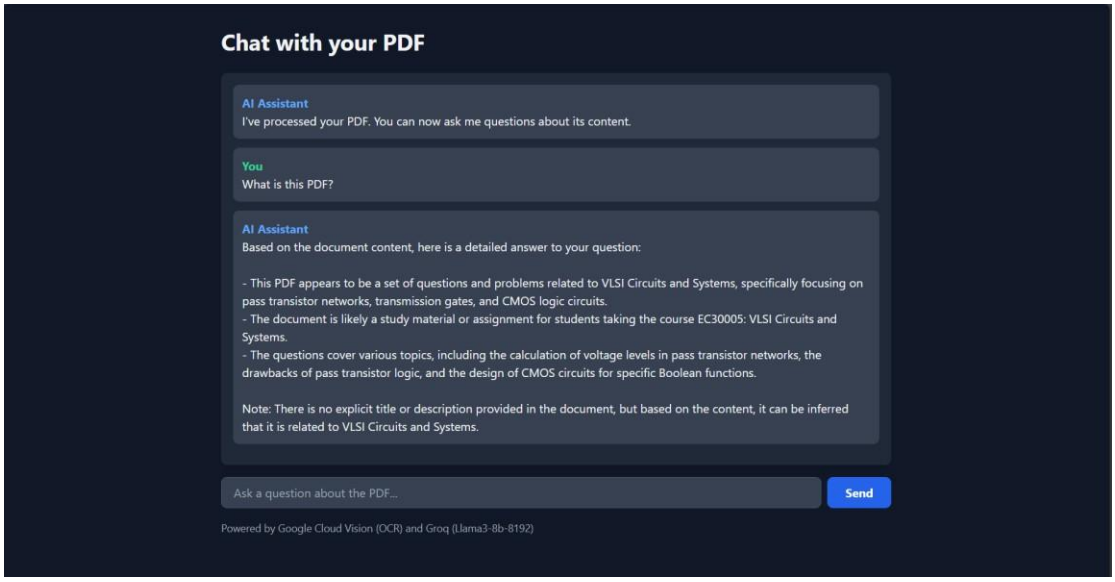
**GitHub Repo ink:** https://github.com/Bhairavg7/AD_LAB_OPEN_END.git

## 4. Output:-

Landing Page:




Chat with PDF page:



**5. Conclusion:** In this experiment, we successfully integrated Python flask backed with frontend Using Vanilla JavaScript, HTML, and CSS, the structure and functionality were implemented effectively, while Tailwind CSS was utilized to streamline styling and create a visually appealing and adaptable user interface. This cohesive combination ensured a seamless experience, enabling users to interact with the application effortlessly.

Bhairav Ganguly

(Name of the Student)

Signature of the Lab Coordinator

(Name of the Coordinator)