LABORATORY REPORT

Application Development Lab
(CS33002)

B.Tech Program in CSE

Submitted By

Name:- Shreeya Das

Roll No: 2205158

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

Spring 2024-2025

Table of Content

Lab Number	8
Experiment Number	6
Experiment Title	Web Scraper using LLMs
Date of Experiment	13/03/2025
Date of Submission	20/03/2025

1. Objective:-

To create a web scraper application integrated with LLMs for processing scraped data.

2. Procedure:- (Steps Followed)

1. Use Python libraries like BeautifulSoup and Requests to scrape web data.

2. You can also use LlamaIndex for Web Scraping and Ollama for open ended LLMs.

3. Integrate LLMs to process and summarize the scraped information.

4. Develop a Flask backend for handling scraping tasks and queries.

5. Create an HTML/CSS frontend to initiate scraping (like the web page to scrape) and display results.

6. You can also take a topic and search the web for a web page and then scrape it.

3. Code:-

index.html U app.py 1, M

app.py > ...

```python
from flask import Flask, request, jsonify
import re
import requests
from bs4 import BeautifulSoup
from youtube_transcript_api import YouTubeTranscriptApi
from duckduckgo_search import DDGS  # Corrected import
import google.generativeai as genai  # Using Gemini API for LLM-based text processing
from dotenv import load_dotenv
import os
from flask_cors import CORS

# Initialize Flask app
app = Flask(__name__)
CORS(app)  # Allow all origins
load_dotenv()  # Load environment variables from .env

# Configure Gemini API
genai.configure(api_key=os.getenv("GEMINI_API_KEY"))

# Function to classify user input
def classify_input(user_text):
    url_pattern = re.compile(r'https?://\S+')
    youtube_pattern = re.compile(r'(https?://)?(www\.)?(youtube\.com|youtu\.?be)/.+')

    urls = url_pattern.findall(user_text)
    youtube_links = [url for url in urls if youtube_pattern.match(url)]

    if youtube_links:
        return {"type": "youtube", "links": youtube_links}
    elif urls:
        return {"type": "web", "links": urls}
    else:
        return {"type": "search", "query": user_text}
```

```python
# Web Scraper
def scrape_website(url):
    try:
        response = requests.get(url, timeout=5)
        if response.status_code != 200:
            return "Failed to retrieve page"

        soup = BeautifulSoup(response.text, 'html.parser')
        paragraphs = soup.find_all('p')
        text = '\n'.join([p.get_text() for p in paragraphs])
        return text[:5000]  # Limit text size
    except Exception as e:
        return f"Scraping error: {str(e)}"

# YouTube Transcript Extractor
def get_youtube_transcript(url):
    video_id = re.search(r"(?:v=|\/)([0-9A-Za-z_-]{11})", url)
    if not video_id:
        return "Invalid YouTube URL"
    try:
        transcript = YouTubeTranscriptApi.get_transcript(video_id.group(1))
        return ' '.join([t['text'] for t in transcript])
    except:
        return "Transcript not available"

# Web Search & Scrape (Improved)
def search_web(query):
    with DDGS() as ddgs:
        results = list(ddgs.text(query, max_results=3))  # Get up to 3 results

    valid_content = []
    for res in results:
        if 'href' in res:
            content = scrape_website(res['href'])
            if content and "Failed" not in content:
                valid_content.append(content)

    if valid_content:
        return "\n".join(valid_content[:2])  # Combine max 2 sources
```

index.html U | app.py 1, M

templates > index.html > html > head > style

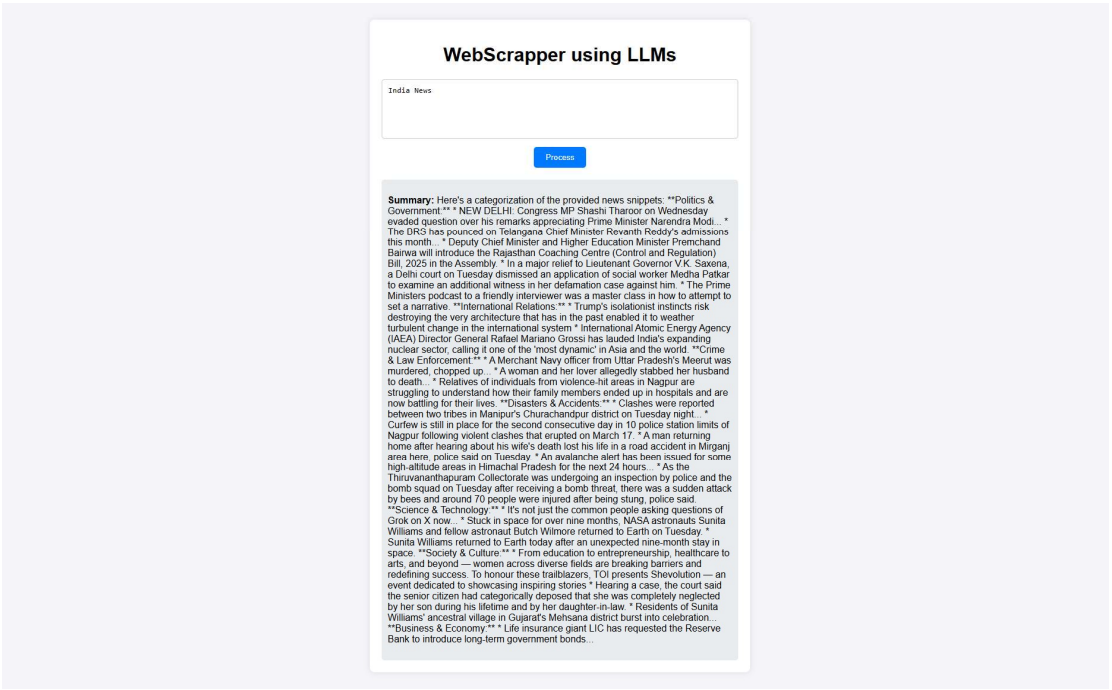
```html
<html lang="en">
    <body>
        <div class="container">
            <h1>Web Scrapper using LLMs</h1>
            <textarea
                id="userInput"
                placeholder="Enter text, URL, or query..."
            ></textarea>
            <button onclick="processInput()">Process</button>
            <div id="result"></div>
        </div>
        <script>
            function processInput() {
                const userInput = document
                    .getElementById("userInput")
                    .value.trim();
                const resultDiv = document.getElementById("result");

                if (!userInput) {
                    resultDiv.innerHTML =
                        "<p style='color: red;'>Please enter some text.</p>";
                    return;
                }

                resultDiv.innerHTML = "<p>Processing...</p>";

                fetch("http://localhost:5000/process", {
                    method: "POST",
                    headers: {
                        "Content-Type": "application/json",
                    },
                    body: JSON.stringify({ text: userInput }),
                })
                    .then((response) => response.json())
                    .then((data) => {
                        resultDiv.innerHTML = `<p><strong>Summary:</strong> ${data.summary}</p>`;
                    })
                    .catch((error) => {
                        resultDiv.innerHTML =
                            "<p style='color: red;'>Error processing request.</p>";
                        console.error("Error:", error);
                    });
```

4. Results/Output:-



5. Remarks:-

Signature of the Student

_____

(Name of the Student)

Signature of the Lab Coordinator

_____

(Name of the Coordinator)