***Bahria University Islamabad Campus***

**Project Report:**

***News Summarizer***

**Course:**

***Natural Language Processing***

**Group Members:**

1. ***Maaz Hussain 01-134221-035***
2. ***M.Sibtain 01-134221-051***
3. ***Abdullah Hassan 01-134221-007***

***Index.html***

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

<meta name="viewport" content="width=device-width, initial-scale=1.0" />

<title>SumNews - Stay Informed</title>

<link

rel="stylesheet"

href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.0/css/all.min.css"

/>

<link rel="stylesheet" href="./assets/css/indexStyle.css" />

</head>

<body>

<!-- Header Section -->

<header>

<div class="container header-content">

<div class="logo">

<i class="fas fa-newspaper"></i>

<span>SumNews</span>

</div>

<nav>

<ul>

<li><a href="#" id="homeLink">Home</a></li>

<li><a href="#" id="dawnLink">Dawn News</a></li>

<li><a href="#" id="tribuneLink">Tribune News</a></li>

<li><a href="#" id="thenewsLink">News International</a></li>

</ul>

</nav>

</div>

</header>

<!-- Hero Section -->

<section class="hero">

<div class="container">

<div class="hero-content">

<h1>Stay Informed with the Latest News</h1>

<p>

Get concise summaries of top news stories from multiple sources.

</p>

<button class="btn btn-primary" id="exploreBtn">

Explore Latest Stories

</button>

</div>

</div>

</section>

<!-- Main Content Section -->

<main>

<div class="container">

<div class="section-header">

<h2 class="section-title" id="sectionTitle">Top Stories</h2>

<div class="refresh-container">

<button class="btn btn-primary" id="refreshBtn">

<i class="fas fa-sync-alt"></i> Refresh News

</button>

</div>

</div>

<div class="spinner" id="loadingSpinner"></div>

<div class="news-grid" id="newsContainer">

<!-- News articles will be loaded here dynamically -->

</div>

</div>

</main>

<!-- Footer Section -->

<footer>

<div class="container">

<div class="footer-content">

<div class="footer-column">

<h3>SumNews</h3>

<p>Your source for concise news summaries from top publications.</p>

<div class="social-links">

<a href="#"><i class="fab fa-facebook-f"></i></a>

<a href="#"><i class="fab fa-twitter"></i></a>

<a href="#"><i class="fab fa-instagram"></i></a>

</div>

</div>

<div class="footer-column">

<h3>Quick Links</h3>

<ul class="footer-links">

<li><a href="#" id="footerHomeLink">Home</a></li>

<li><a href="#" id="footerDawnLink">Dawn News</a></li>

<li><a href="#" id="footerTribuneLink">Tribune News</a></li>

<li><a href="#" id="footerThenewsLink">News International</a></li>

</ul>

</div>

<div class="footer-column">

<h3>Contact</h3>

<ul class="footer-links">

<li><a href="mailto:contact@sumnews.com">hi@sumnews.com</a></li>

<li><a href="#">About Us</a></li>

<li><a href="#">Privacy Policy</a></li>

</ul>

</div>

</div>

<div class="copyright">

<p>&copy; 2025 SumNews. All rights reserved.</p>

</div>

</div>

</footer>

<script src="./assets/js/indexHTML.js"></script>

</body>

</html>

***IndexStyle.css***

/\* Global Styles \*/

:root {

--primary: #2563eb;

--primary-dark: #1d4ed8;

--secondary: #e11d48;

--dark: #1e293b;

--light: #f8fafc;

--gray: #94a3b8;

--dark-gray: #64748b;

}

\* {

margin: 0;

padding: 0;

box-sizing: border-box;

}

body {

font-family: "Inter", -apple-system, BlinkMacSystemFont, sans-serif;

line-height: 1.6;

color: var(--dark);

background-color: var(--light);

min-height: 100vh;

display: flex;

flex-direction: column;

}

.container {

max-width: 1200px;

margin: 0 auto;

padding: 0 20px;

}

.btn {

display: inline-block;

padding: 10px 20px;

border-radius: 6px;

font-weight: 600;

cursor: pointer;

transition: all 0.3s ease;

text-decoration: none;

text-align: center;

}

.btn-primary {

background-color: var(--primary);

color: white;

border: none;

}

.btn-primary:hover {

background-color: var(--primary-dark);

transform: translateY(-2px);

}

.btn-outline {

background-color: transparent;

color: var(--primary);

border: 2px solid var(--primary);

}

.btn-outline:hover {

background-color: var(--primary);

color: white;

}

/\* Header Styles \*/

header {

background-color: white;

box-shadow: 0 2px 10px rgba(0, 0, 0, 0.1);

position: sticky;

top: 0;

z-index: 100;

}

.header-content {

display: flex;

justify-content: space-between;

align-items: center;

padding: 15px 0;

}

.logo {

font-size: 28px;

font-weight: 800;

color: var(--primary);

display: flex;

align-items: center;

gap: 10px;

}

.logo i {

font-size: 32px;

}

nav ul {

display: flex;

list-style: none;

gap: 25px;

}

nav ul li a {

color: var(--dark);

text-decoration: none;

font-weight: 600;

font-size: 16px;

transition: color 0.3s;

position: relative;

}

nav ul li a:hover {

color: var(--primary);

}

nav ul li a::after {

content: "";

position: absolute;

width: 0;

height: 2px;

bottom: -5px;

left: 0;

background-color: var(--primary);

transition: width 0.3s;

}

nav ul li a:hover::after {

width: 100%;

}

/\* Hero Section \*/

.hero {

background: linear-gradient(135deg, var(--primary), var(--primary-dark));

color: white;

padding: 80px 0;

margin-bottom: 40px;

}

.hero-content {

max-width: 600px;

}

.hero h1 {

font-size: 48px;

margin-bottom: 20px;

line-height: 1.2;

}

.hero p {

font-size: 18px;

margin-bottom: 30px;

opacity: 0.9;

}

/\* Main Content Styles \*/

main {

flex: 1;

padding: 40px 0;

}

.section-header {

display: flex;

justify-content: space-between;

align-items: center;

margin-bottom: 30px;

}

.section-title {

font-size: 32px;

font-weight: 700;

color: var(--dark);

position: relative;

}

.section-title::after {

content: "";

position: absolute;

width: 50px;

height: 4px;

bottom: -10px;

left: 0;

background-color: var(--primary);

}

.view-all {

color: var(--primary);

font-weight: 600;

text-decoration: none;

display: flex;

align-items: center;

gap: 5px;

}

.news-grid {

display: grid;

grid-template-columns: repeat(auto-fill, minmax(350px, 1fr));

gap: 30px;

}

.news-article {

background-color: white;

border-radius: 12px;

overflow: hidden;

box-shadow: 0 5px 15px rgba(0, 0, 0, 0.05);

transition: transform 0.3s, box-shadow 0.3s;

display: flex;

flex-direction: column;

height: 100%;

}

.news-article:hover {

transform: translateY(-10px);

box-shadow: 0 15px 30px rgba(0, 0, 0, 0.1);

}

.article-content {

padding: 25px;

flex: 1;

display: flex;

flex-direction: column;

}

.source-label {

display: inline-block;

background-color: var(--dark-gray);

color: white;

padding: 4px 12px;

border-radius: 20px;

font-size: 12px;

font-weight: 600;

margin-bottom: 15px;

text-transform: uppercase;

align-self: flex-start;

}

.source-dawn {

background-color: #3b82f6;

}

.source-thenews {

background-color: #ef4444;

}

.source-tribune {

background-color: #10b981;

}

.news-heading {

color: var(--dark);

font-size: 20px;

margin-bottom: 15px;

line-height: 1.4;

}

.news-summary {

color: var(--dark-gray);

line-height: 1.6;

margin-bottom: 20px;

flex: 1;

}

.article-footer {

display: flex;

justify-content: space-between;

align-items: center;

margin-top: auto;

}

.publish-date {

font-size: 14px;

color: var(--gray);

}

.read-more {

color: var(--primary);

font-weight: 600;

text-decoration: none;

font-size: 14px;

display: flex;

align-items: center;

gap: 5px;

}

.refresh-container {

text-align: center;

margin: 30px 0;

}

/\* Loading Spinner \*/

.spinner {

display: none;

width: 40px;

height: 40px;

margin: 30px auto;

border: 4px solid rgba(0, 0, 0, 0.1);

border-radius: 50%;

border-top: 4px solid var(--primary);

animation: spin 1s linear infinite;

}

@keyframes spin {

0% {

transform: rotate(0deg);

}

100% {

transform: rotate(360deg);

}

}

/\* Footer Styles \*/

footer {

background-color: var(--dark);

color: white;

padding: 60px 0 30px;

}

.footer-content {

display: grid;

grid-template-columns: repeat(auto-fit, minmax(200px, 1fr));

gap: 40px;

margin-bottom: 40px;

}

.footer-column h3 {

font-size: 18px;

margin-bottom: 20px;

position: relative;

padding-bottom: 10px;

}

.footer-column h3::after {

content: "";

position: absolute;

width: 40px;

height: 3px;

bottom: 0;

left: 0;

background-color: var(--primary);

}

.footer-links {

list-style: none;

}

.footer-links li {

margin-bottom: 10px;

}

.footer-links a {

color: var(--gray);

text-decoration: none;

transition: color 0.3s;

}

.footer-links a:hover {

color: white;

}

.social-links {

display: flex;

gap: 15px;

margin-top: 20px;

}

.social-links a {

display: flex;

align-items: center;

justify-content: center;

width: 40px;

height: 40px;

border-radius: 50%;

background-color: rgba(255, 255, 255, 0.1);

color: white;

transition: background-color 0.3s;

}

.social-links a:hover {

background-color: var(--primary);

}

.copyright {

text-align: center;

padding-top: 30px;

border-top: 1px solid rgba(255, 255, 255, 0.1);

color: var(--gray);

font-size: 14px;

}

/\* Responsive Styles \*/

@media (max-width: 992px) {

.hero h1 {

font-size: 36px;

}

}

@media (max-width: 768px) {

.header-content {

flex-direction: column;

gap: 20px;

}

nav ul {

flex-wrap: wrap;

justify-content: center;

}

.hero {

text-align: center;

padding: 60px 0;

}

.hero-content {

margin: 0 auto;

}

.section-header {

flex-direction: column;

align-items: flex-start;

gap: 15px;

}

}

@media (max-width: 576px) {

.news-grid {

grid-template-columns: 1fr;

}

}

***IndexHTML.js***

// API endpoints

const API\_ENDPOINTS = {

dawn: "http://localhost:5002/api/dawn/latest?limit=10",

thenews: "http://localhost:5003/api/thenews/latest?limit=10",

tribune: "http://localhost:5001/api/tribune/latest?limit=10",

};

// DOM elements

const newsContainer = document.getElementById("newsContainer");

const refreshBtn = document.getElementById("refreshBtn");

const exploreBtn = document.getElementById("exploreBtn");

const loadingSpinner = document.getElementById("loadingSpinner");

const sectionTitle = document.getElementById("sectionTitle");

// Navigation links

const homeLink = document.getElementById("homeLink");

const dawnLink = document.getElementById("dawnLink");

const tribuneLink = document.getElementById("tribuneLink");

const thenewsLink = document.getElementById("thenewsLink");

// Footer links

const footerHomeLink = document.getElementById("footerHomeLink");

const footerDawnLink = document.getElementById("footerDawnLink");

const footerTribuneLink = document.getElementById("footerTribuneLink");

const footerThenewsLink = document.getElementById("footerThenewsLink");

// Current state

let currentFilter = "all";

let allArticles = [];

// Function to handle article clicks

function handleArticleClick(article) {

localStorage.setItem("currentArticle", JSON.stringify(article));

window.open("article-detail.html", "\_blank");

}

// Fetch news from all sources

async function fetchAllNews() {

try {

loadingSpinner.style.display = "block";

newsContainer.innerHTML = "";

if (allArticles.length === 0 || currentFilter === "refresh") {

const [dawnResponse, thenewsResponse, tribuneResponse] =

await Promise.all([

fetch(API\_ENDPOINTS.dawn).then((res) => res.json()),

fetch(API\_ENDPOINTS.thenews).then((res) => res.json()),

fetch(API\_ENDPOINTS.tribune).then((res) => res.json()),

]);

// Reset all articles

allArticles = [];

// Process Dawn articles

if (dawnResponse.status === "success") {

dawnResponse.articles.forEach((article) => {

article.source = "dawn";

allArticles.push(article);

});

}

// Process The News articles

if (thenewsResponse.status === "success") {

thenewsResponse.articles.forEach((article) => {

article.source = "thenews";

allArticles.push(article);

});

}

// Process Tribune articles

if (tribuneResponse.status === "success") {

tribuneResponse.articles.forEach((article) => {

article.source = "tribune";

allArticles.push(article);

});

}

// Sort all articles by date (newest first)

allArticles.sort((a, b) => {

return new Date(b.published\_date) - new Date(a.published\_date);

});

}

filterAndDisplayArticles();

} catch (error) {

console.error("Error fetching news:", error);

showError("Failed to load news. Please try again later.");

} finally {

loadingSpinner.style.display = "none";

if (currentFilter === "refresh") {

currentFilter = "all";

}

}

}

// Filter articles based on current filter and display them

function filterAndDisplayArticles() {

try {

updateSectionTitle();

let filteredArticles = allArticles;

if (currentFilter === "dawn") {

filteredArticles = allArticles.filter(

(article) => article.source === "dawn"

);

} else if (currentFilter === "thenews") {

filteredArticles = allArticles.filter(

(article) => article.source === "thenews"

);

} else if (currentFilter === "tribune") {

filteredArticles = allArticles.filter(

(article) => article.source === "tribune"

);

}

newsContainer.innerHTML = "";

if (filteredArticles.length > 0) {

filteredArticles.forEach((article) => {

try {

if (article.source === "dawn") {

addNewsArticle(article, "Dawn", "source-dawn");

} else if (article.source === "thenews") {

addNewsArticle(article, "The News", "source-thenews");

} else if (article.source === "tribune") {

addNewsArticle(article, "Tribune", "source-tribune");

}

} catch (error) {

console.error("Error rendering article:", error);

}

});

} else {

showError("No news articles found for this source.");

}

} catch (error) {

console.error("Error filtering articles:", error);

showError("Error displaying news. Please try again.");

}

}

// Update section title based on current filter

function updateSectionTitle() {

if (currentFilter === "all") {

sectionTitle.textContent = "Top Stories";

} else if (currentFilter === "dawn") {

sectionTitle.textContent = "Dawn News";

} else if (currentFilter === "thenews") {

sectionTitle.textContent = "The News International";

} else if (currentFilter === "tribune") {

sectionTitle.textContent = "Tribune News";

}

}

// Show error message

function showError(message) {

newsContainer.innerHTML = `

<div class="error-message" style="grid-column: 1/-1; text-align: center; padding: 20px;">

<p>${message}</p>

</div>

`;

}

// Add a news article to the DOM

function addNewsArticle(article, source, sourceClass) {

if (!article || !article.title) {

console.warn("Skipping invalid article:", article);

return;

}

const articleElement = document.createElement("article");

articleElement.className = "news-article";

try {

let date = article.published\_date || "Date not available";

let description =

article.summary || article.content || "No description available";

if (description.length > 150) {

description = description.substring(0, 150) + "...";

}

articleElement.innerHTML = `

<div class="article-content">

<span class="source-label ${sourceClass}">${source}</span>

<h3 class="news-heading">${article.title}</h3>

<p class="news-summary">${description}</p>

<div class="article-footer">

<span class="publish-date">${date}</span>

<a href="#" class="read-more">Read More <i class="fas fa-arrow-right"></i></a>

</div>

</div>

`;

articleElement.addEventListener("click", () => handleArticleClick(article));

const readMoreLink = articleElement.querySelector(".read-more");

readMoreLink.addEventListener("click", (e) => {

e.preventDefault();

e.stopPropagation();

handleArticleClick(article);

});

newsContainer.appendChild(articleElement);

} catch (error) {

console.error("Error creating article element:", error);

articleElement.innerHTML = `

<div class="article-content">

<span class="source-label ${sourceClass}">${source}</span>

<h3 class="news-heading">${

article.title || "News Article"

}</h3>

<p class="news-summary">Click to read more about this story</p>

<div class="article-footer">

<span class="publish-date">Today</span>

<a href="#" class="read-more">Read More <i class="fas fa-arrow-right"></i></a>

</div>

</div>

`;

articleElement.addEventListener("click", () => handleArticleClick(article));

newsContainer.appendChild(articleElement);

}

}

// Event Listeners

refreshBtn.addEventListener("click", () => {

currentFilter = "refresh";

fetchAllNews();

});

exploreBtn.addEventListener("click", () => {

currentFilter = "all";

fetchAllNews();

});

homeLink.addEventListener("click", (e) => {

e.preventDefault();

currentFilter = "all";

fetchAllNews();

});

dawnLink.addEventListener("click", (e) => {

e.preventDefault();

currentFilter = "dawn";

fetchAllNews();

});

thenewsLink.addEventListener("click", (e) => {

e.preventDefault();

currentFilter = "thenews";

fetchAllNews();

});

tribuneLink.addEventListener("click", (e) => {

e.preventDefault();

currentFilter = "tribune";

fetchAllNews();

});

// Footer link events

footerHomeLink.addEventListener("click", (e) => {

e.preventDefault();

currentFilter = "all";

fetchAllNews();

});

footerDawnLink.addEventListener("click", (e) => {

e.preventDefault();

currentFilter = "dawn";

fetchAllNews();

});

footerThenewsLink.addEventListener("click", (e) => {

e.preventDefault();

currentFilter = "thenews";

fetchAllNews();

});

footerTribuneLink.addEventListener("click", (e) => {

e.preventDefault();

currentFilter = "tribune";

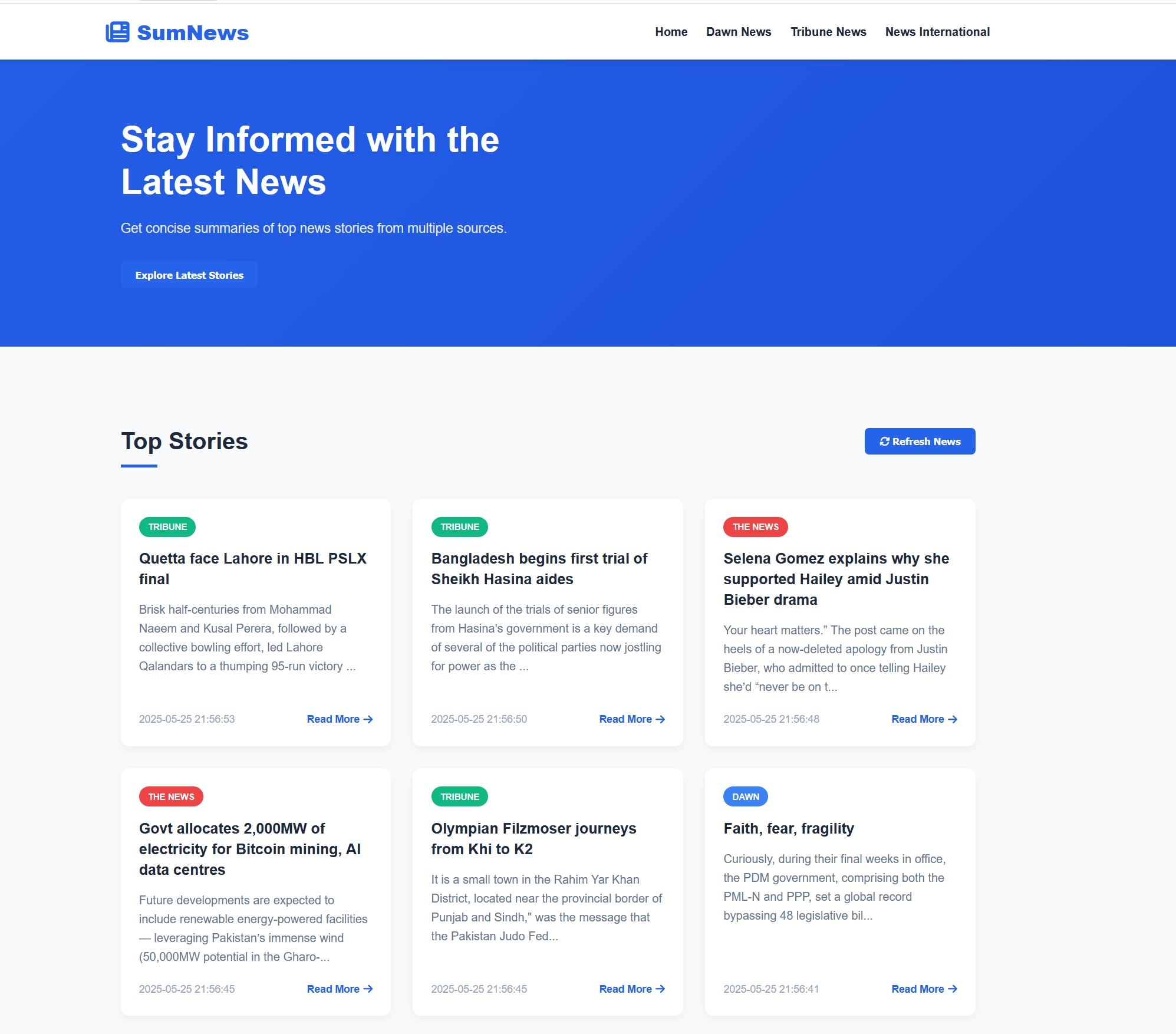
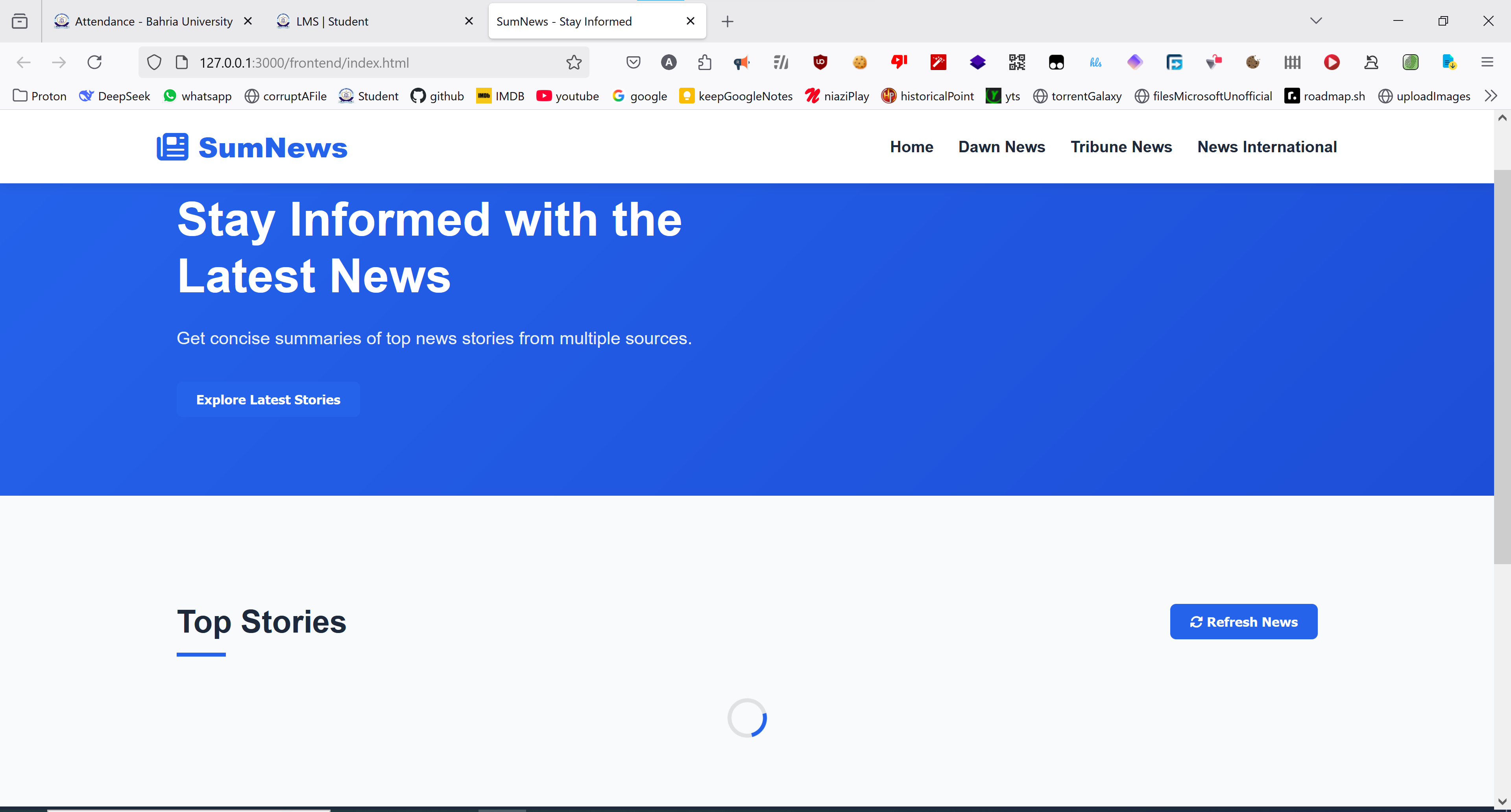
fetchAllNews();

});

// Load news when page loads

document.addEventListener("DOMContentLoaded", fetchAllNews);

***Output:***



***Article-Details.html***

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

<meta name="viewport" content="width=device-width, initial-scale=1.0" />

<title>Article Details - SumNews</title>

<link

rel="stylesheet"

href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.0/css/all.min.css"

/>

<link rel="stylesheet" href="./assets/css/article-detail.css" />

</head>

<body>

<!-- Header Section -->

<header>

<div class="container header-content">

<a href="index.html" class="logo">

<i class="fas fa-newspaper"></i>

<span>SumNews</span>

</a>

<a href="index.html" class="back-btn">

<i class="fas fa-arrow-left"></i>

Back to News

</a>

</div>

</header>

<!-- Article Detail Section -->

<main class="container">

<article class="article-detail" id="articleDetail">

<!-- Article content will be loaded here dynamically -->

</article>

</main>

<script src="./assets/js/article-detail.js"></script>

</body>

</html>

***article-detail.css***

:root {

--primary: #2563eb;

--primary-dark: #1d4ed8;

--dark: #1e293b;

--light: #f8fafc;

--gray: #94a3b8;

--light-gray: #e2e8f0;

}

\* {

margin: 0;

padding: 0;

box-sizing: border-box;

}

body {

font-family: "Inter", -apple-system, BlinkMacSystemFont, sans-serif;

line-height: 1.6;

color: var(--dark);

background-color: var(--light);

min-height: 100vh;

}

.container {

max-width: 1200px;

margin: 0 auto;

padding: 0 20px;

}

header {

background-color: white;

box-shadow: 0 2px 10px rgba(0, 0, 0, 0.1);

position: sticky;

top: 0;

z-index: 100;

}

.header-content {

display: flex;

justify-content: space-between;

align-items: center;

padding: 15px 0;

}

.logo {

font-size: 28px;

font-weight: 800;

color: var(--primary);

display: flex;

align-items: center;

gap: 10px;

text-decoration: none;

}

.back-btn {

display: inline-flex;

align-items: center;

gap: 8px;

color: var(--primary);

font-weight: 600;

text-decoration: none;

padding: 10px 15px;

border-radius: 6px;

transition: background-color 0.3s;

}

.back-btn:hover {

background-color: rgba(37, 99, 235, 0.1);

}

.article-detail {

background-color: white;

border-radius: 12px;

box-shadow: 0 5px 15px rgba(0, 0, 0, 0.05);

padding: 40px;

margin: 40px 0;

}

.article-header {

margin-bottom: 20px;

}

.source-label {

display: inline-block;

background-color: var(--gray);

color: white;

padding: 4px 12px;

border-radius: 20px;

font-size: 12px;

font-weight: 600;

margin-bottom: 15px;

text-transform: uppercase;

}

.source-dawn {

background-color: #3b82f6;

}

.source-thenews {

background-color: #ef4444;

}

.source-tribune {

background-color: #10b981;

}

.article-title {

font-size: 32px;

margin-bottom: 15px;

line-height: 1.3;

}

.article-meta {

display: flex;

align-items: center;

gap: 15px;

color: var(--gray);

font-size: 14px;

margin-bottom: 15px;

}

.original-link-top {

display: inline-flex;

align-items: center;

gap: 8px;

color: var(--primary);

font-weight: 600;

text-decoration: none;

margin-bottom: 20px;

padding: 8px 15px;

border: 1px solid var(--primary);

border-radius: 6px;

transition: all 0.3s;

}

.original-link-top:hover {

background-color: rgba(37, 99, 235, 0.1);

transform: translateY(-2px);

}

.article-image {

width: 100%;

max-height: 500px;

object-fit: cover;

border-radius: 8px;

margin-bottom: 30px;

}

.summary-section {

background-color: var(--light-gray);

padding: 25px;

border-radius: 8px;

margin-bottom: 30px;

}

.summary-title {

font-size: 20px;

margin-bottom: 15px;

color: var(--primary);

display: flex;

align-items: center;

gap: 10px;

}

.summary-content {

font-size: 16px;

line-height: 1.7;

}

.article-content {

font-size: 18px;

line-height: 1.8;

}

.article-content p {

margin-bottom: 20px;

}

.original-link-bottom {

display: inline-flex;

align-items: center;

gap: 8px;

color: var(--primary);

font-weight: 600;

text-decoration: none;

margin-top: 30px;

padding: 10px 15px;

border-radius: 6px;

transition: background-color 0.3s;

}

.original-link-bottom:hover {

background-color: rgba(37, 99, 235, 0.1);

}

@media (max-width: 768px) {

.article-detail {

padding: 25px;

}

.article-title {

font-size: 24px;

}

.article-content {

font-size: 16px;

}

}

***article-detail.js***

document.addEventListener("DOMContentLoaded", function () {

const articleDetail = document.getElementById("articleDetail");

// Get the article data from localStorage

const articleData = JSON.parse(localStorage.getItem("currentArticle"));

if (!articleData) {

articleDetail.innerHTML = `

<div class="error-message">

<h2>Article Not Found</h2>

<p>We couldn't find the article you're looking for.</p>

<a href="index.html" class="original-link-top">

<i class="fas fa-arrow-left"></i>

Back to News

</a>

</div>

`;

return;

}

// Determine source class

let sourceClass = "";

let sourceName = "";

if (articleData.source === "dawn") {

sourceClass = "source-dawn";

sourceName = "Dawn News";

} else if (articleData.source === "thenews") {

sourceClass = "source-thenews";

sourceName = "The News";

} else if (articleData.source === "tribune") {

sourceClass = "source-tribune";

sourceName = "Tribune News";

}

// Format date

let date = "";

if (articleData.published\_date) {

date = new Date(articleData.published\_date).toLocaleDateString("en-US", {

year: "numeric",

month: "long",

day: "numeric",

hour: "2-digit",

minute: "2-digit",

});

} else {

date = "Date not available";

}

// Create the article detail HTML with summary section

articleDetail.innerHTML = `

<div class="article-header">

<span class="source-label ${sourceClass}">${sourceName}</span>

<h1 class="article-title">${

articleData.title || "No title available"

}</h1>

<div class="article-meta">

<span>${date}</span>

<span>${

articleData.word\_count

? articleData.word\_count + " words"

: ""

}</span>

</div>

</div>

${

articleData.url

? `

<a href="${articleData.url}" target="\_blank" rel="noopener noreferrer" class="original-link-top">

<i class="fas fa-external-link-alt"></i>

Read Original Article

</a>

`

: ""

}

${

articleData.image\_url

? `<img src="${articleData.image\_url}" alt="${articleData.title}" class="article-image">`

: ""

}

<div class="summary-section">

<h3 class="summary-title">

<i class="fas fa-file-contract"></i>

Summary

</h3>

<div class="summary-content">

${

articleData.summary ||

"No summary available for this article."

}

</div>

</div>

<div class="article-content">

<h3>Full Article</h3>

${

articleData.content

? `<p>${articleData.content.replace(

/\n/g,

"</p><p>"

)}</p>`

: articleData.description

? `<p>${articleData.description}</p>`

: "<p>No content available for this article.</p>"

}

</div>

${

articleData.url

? `

<a href="${articleData.url}" target="\_blank" rel="noopener noreferrer" class="original-link-bottom">

<i class="fas fa-external-link-alt"></i>

Continue Reading on ${sourceName}

</a>

`

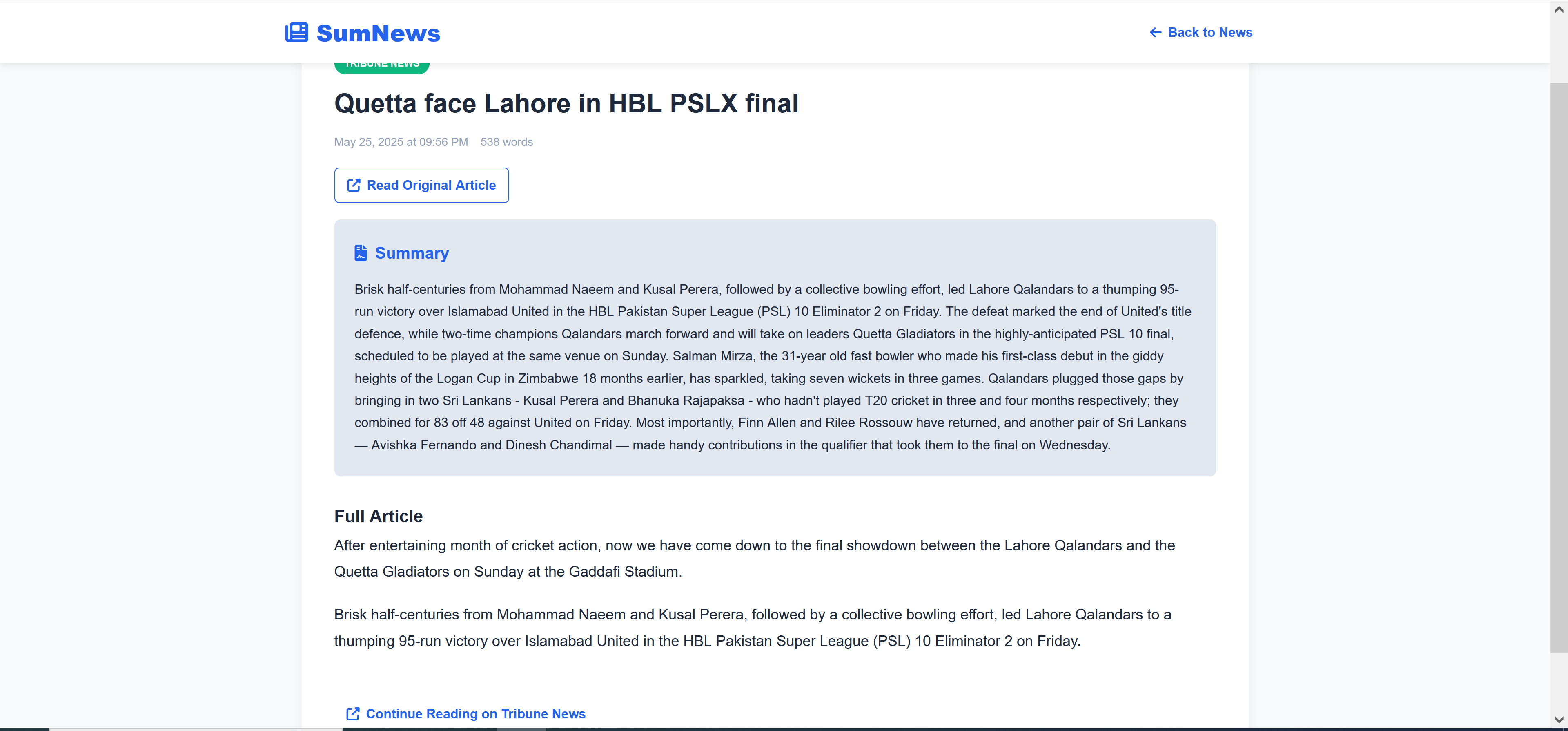
: ""

}

`;

});

***Output:***



***Dawn News Api***

from flask import Flask, jsonify, request

from flask\_cors import CORS

import requests

from bs4 import BeautifulSoup

import time

import random

from algorithm.summarization import summarize\_tfidf

# Flask app setup

app = Flask(\_\_name\_\_)

CORS(app)

# ---------------------- SCRAPING FUNCTIONS ----------------------

def get\_dawn\_news\_links():

headers = {

'User-Agent': 'Mozilla/5.0'

}

try:

response = requests.get('https://www.dawn.com', headers=headers, timeout=15)

soup = BeautifulSoup(response.text, 'html.parser')

links = []

for link in soup.find\_all('a', href=True):

href = link['href']

if '/news/' in href and href not in links:

if not href.startswith('http'):

href = f'https://www.dawn.com{href}'

links.append(href)

return links

except Exception as e:

print(f"Error getting links: {e}")

return []

def scrape\_dawn\_article(url):

headers = {

'User-Agent': 'Mozilla/5.0'

}

try:

response = requests.get(url, headers=headers, timeout=15)

soup = BeautifulSoup(response.text, 'html.parser')

title = soup.find('meta', property='og:title')

if not title:

return None

content = []

article\_body = soup.find('div', class\_='story\_\_content')

if article\_body:

for tag in article\_body(['script', 'style', 'figure']):

tag.decompose()

for p in article\_body.find\_all('p'):

text = p.get\_text(strip=True)

if text:

content.append(text)

full\_text = '\n\n'.join(content)

summary\_sentences = summarize\_tfidf(full\_text, number\_of\_sentences=5) # Use TF-IDF summarization

return {

'title': title['content'],

'url': url,

'content': full\_text,

'summary': ' '.join(summary\_sentences),

'word\_count': len(full\_text.split()),

'published\_date': time.strftime('%Y-%m-%d %H:%M:%S') # Add current time as published date

}

except Exception as e:

print(f"Error scraping {url}: {e}")

return {

'title': 'Failed to scrape',

'url': url,

'content': 'Error occurred while scraping this article',

'summary': '',

'word\_count': 0,

'published\_date': time.strftime('%Y-%m-%d %H:%M:%S')

}

# ---------------------- API ENDPOINTS ----------------------

@app.route('/api/dawn/latest', methods=['GET'])

def get\_latest\_dawn\_articles():

try:

limit = int(request.args.get('limit', 5))

links = get\_dawn\_news\_links()

random.shuffle(links)

selected\_links = links[:limit]

articles = []

for i, url in enumerate(selected\_links):

print(f"Scraping {i+1}/{limit}: {url}")

article = scrape\_dawn\_article(url)

if article:

articles.append(article)

time.sleep(1)

return jsonify({

'status': 'success',

'count': len(articles),

'articles': articles

})

except Exception as e:

return jsonify({'status': 'error', 'message': str(e)}), 500

# ---------------------- MAIN ----------------------

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True, port=5002)

***The News Api***

from flask import Flask, jsonify, request

from flask\_cors import CORS

import requests

from bs4 import BeautifulSoup

import time

import random

from algorithm.summarization import summarize\_tfidf

# Flask app setup

app = Flask(\_\_name\_\_)

CORS(app)

# ---------------------- SCRAPING FUNCTIONS ----------------------

def get\_thenews\_latest\_links():

headers = {

'User-Agent': 'Mozilla/5.0'

}

try:

response = requests.get('https://www.thenews.com.pk/', headers=headers, timeout=15)

soup = BeautifulSoup(response.text, 'html.parser')

latest\_links = set()

for link in soup.find\_all('a', href=True):

href = link['href']

if '/latest/' in href and not href.startswith('javascript'):

if not href.startswith('http'):

href = f'https://www.thenews.com.pk{href}'

latest\_links.add(href)

return list(latest\_links)

except Exception as e:

print(f"Error getting The News links: {e}")

return []

def scrape\_thenews\_article(url):

headers = {

'User-Agent': 'Mozilla/5.0'

}

try:

response = requests.get(url, headers=headers, timeout=15)

soup = BeautifulSoup(response.text, 'html.parser')

title\_tag = soup.find('h1', class\_='detail-title') or soup.find('h1')

title = title\_tag.get\_text(strip=True) if title\_tag else 'No title found'

content = []

article\_container = (soup.find('div', class\_='detail-story') or

soup.find('article') or

soup.find('div', class\_='story\_\_content') or

soup.find('div', class\_='story-detail'))

if article\_container:

for element in article\_container(['script', 'style', 'figure', 'div.social-share',

'div.related-news', 'div.tags', 'div.author',

'header', 'footer', 'aside', 'iframe']):

element.decompose()

for p in article\_container.find\_all('p'):

text = p.get\_text(strip=True)

if text and len(text) > 30:

if not any(text.startswith(prefix) for prefix in

['Also Read', 'Also read', 'READ MORE', 'Read more', 'Published', 'Updated']):

content.append(text)

full\_text = '\n\n'.join(content)

summary\_sentences = summarize\_tfidf(full\_text, number\_of\_sentences=5) # Use TF-IDF summarization

return {

'title': title,

'url': url,

'content': full\_text,

'summary': ' '.join(summary\_sentences),

'word\_count': len(full\_text.split()),

'published\_date': time.strftime('%Y-%m-%d %H:%M:%S')

}

except Exception as e:

print(f"Error scraping {url}: {e}")

return {

'title': 'Failed to scrape',

'url': url,

'content': 'Error occurred while scraping this article',

'summary': '',

'word\_count': 0,

'published\_date': time.strftime('%Y-%m-%d %H:%M:%S')

}

# ---------------------- API ENDPOINTS ----------------------

@app.route('/api/thenews/latest', methods=['GET'])

def get\_latest\_thenews\_articles():

try:

limit = int(request.args.get('limit', 5))

links = get\_thenews\_latest\_links()

random.shuffle(links)

selected\_links = links[:limit]

articles = []

for i, url in enumerate(selected\_links):

print(f"Scraping {i+1}/{limit}: {url}")

article = scrape\_thenews\_article(url)

if article:

articles.append(article)

time.sleep(1)

return jsonify({

'status': 'success',

'count': len(articles),

'articles': articles

})

except Exception as e:

return jsonify({'status': 'error', 'message': str(e)}), 500

# ---------------------- MAIN ----------------------

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True, port=5003)

***Tribune News Api***

from flask import Flask, jsonify, request

from flask\_cors import CORS

import requests

from bs4 import BeautifulSoup

import time

import random

from algorithm.summarization import summarize\_tfidf

# Flask app setup

app = Flask(\_\_name\_\_)

CORS(app)

# ---------------------- SCRAPING FUNCTIONS ----------------------

def get\_tribune\_latest\_links():

headers = {

'User-Agent': 'Mozilla/5.0'

}

try:

response = requests.get('https://tribune.com.pk/', headers=headers, timeout=15)

soup = BeautifulSoup(response.text, 'html.parser')

story\_links = set()

for link in soup.find\_all('a', href=True):

href = link['href']

if '/story/' in href and not href.startswith('javascript'):

if not href.startswith('http'):

href = f'https://tribune.com.pk{href}'

story\_links.add(href)

return list(story\_links)

except Exception as e:

print(f"Error getting Tribune links: {e}")

return []

def scrape\_tribune\_article(url):

headers = {

'User-Agent': 'Mozilla/5.0'

}

try:

response = requests.get(url, headers=headers, timeout=15)

soup = BeautifulSoup(response.text, 'html.parser')

# Initialize with default values

article\_data = {

'title': 'No title found',

'url': url,

'content': 'Content not available',

'word\_count': 0,

'published\_date': time.strftime('%Y-%m-%d %H:%M:%S')

}

# Try both formats of Tribune articles

story\_box = soup.find('div', class\_='story-box-section')

if story\_box:

# Format 1 processing

title = story\_box.find('h1')

if title:

article\_data['title'] = title.get\_text(strip=True)

story\_text = story\_box.find('span', class\_='story-text')

if story\_text:

for element in story\_text(['script', 'style', 'figure', 'div', 'aside', 'iframe']):

element.decompose()

paragraphs = []

for p in story\_text.find\_all('p'):

text = p.get\_text(strip=True)

if text and len(text) > 30:

paragraphs.append(text)

if paragraphs:

article\_data['content'] = '\n\n'.join(paragraphs)

article\_data['word\_count'] = len(' '.join(paragraphs).split())

else:

# Format 2 processing

title = (soup.find('h1', class\_='story-title') or

soup.find('h1', class\_='title') or

soup.find('h1'))

if title:

article\_data['title'] = title.get\_text(strip=True)

article\_body = (soup.find('div', class\_='story-content') or

soup.find('article') or

soup.find('div', class\_='content-area') or

soup.find('span', class\_='story-text'))

if article\_body:

for element in article\_body(['script', 'style', 'figure', 'div.social-share',

'div.related-posts', 'div.tags', 'div.author',

'header', 'footer', 'aside', 'iframe']):

element.decompose()

paragraphs = []

for p in article\_body.find\_all('p'):

text = p.get\_text(strip=True)

if text and len(text) > 30:

paragraphs.append(text)

if paragraphs:

article\_data['content'] = '\n\n'.join(paragraphs)

article\_data['word\_count'] = len(' '.join(paragraphs).split())

# Generate summary using TF-IDF

if article\_data['content'] != 'Content not available':

summary\_sentences = summarize\_tfidf(article\_data['content'], number\_of\_sentences=5)

article\_data['summary'] = ' '.join(summary\_sentences)

else:

article\_data['summary'] = 'No summary available'

return article\_data

except Exception as e:

print(f"Error scraping {url}: {e}")

return {

'title': 'Failed to scrape',

'url': url,

'content': 'Error occurred while scraping this article',

'summary': '',

'word\_count': 0,

'published\_date': time.strftime('%Y-%m-%d %H:%M:%S')

}

# ---------------------- API ENDPOINTS ----------------------

@app.route('/api/tribune/latest', methods=['GET'])

def get\_latest\_tribune\_articles():

try:

limit = int(request.args.get('limit', 5))

links = get\_tribune\_latest\_links()

random.shuffle(links)

selected\_links = links[:limit]

articles = []

for i, url in enumerate(selected\_links):

print(f"Scraping {i+1}/{limit}: {url}")

article = scrape\_tribune\_article(url)

if article:

articles.append(article)

time.sleep(1)

return jsonify({

'status': 'success',

'count': len(articles),

'articles': articles

})

except Exception as e:

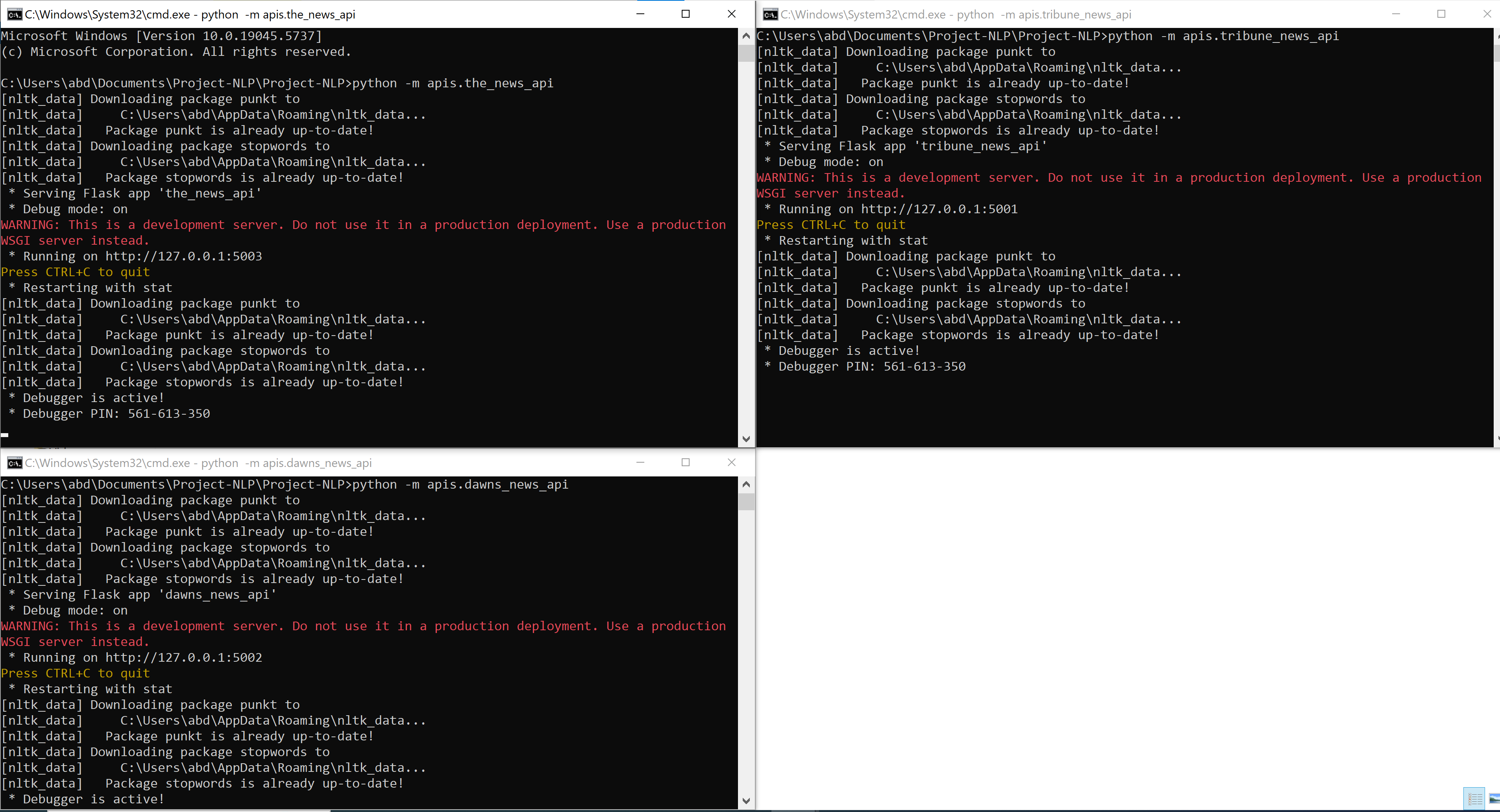
return jsonify({'status': 'error', 'message': str(e)}), 500

# ---------------------- MAIN ----------------------

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True, port=5001)

***Output***



***Summaizer TF-IDF***

import nltk

import string

import numpy as np

from nltk.cluster.util import cosine\_distance

from sklearn.feature\_extraction.text import TfidfVectorizer

nltk.download('punkt')

nltk.download('stopwords')

stopwords = nltk.corpus.stopwords.words('english')

# Preprocessing function

def preprocess(text):

formatted\_text = text.lower()

tokens = [word for word in nltk.word\_tokenize(formatted\_text)]

tokens = [word for word in tokens if word not in stopwords and word not in string.punctuation]

return ' '.join(tokens)

# Calculate sentence importance using TF-IDF

def calculate\_sentence\_tfidf(sentences):

# Create a TF-IDF Vectorizer

vectorizer = TfidfVectorizer(stop\_words='english')

tfidf\_matrix = vectorizer.fit\_transform(sentences)

# Get the importance score for each sentence (sum of TF-IDF scores for words in each sentence)

sentence\_scores = np.array(tfidf\_matrix.sum(axis=1)).flatten()

return sentence\_scores

# Summarization using TF-IDF (No PageRank)

def summarize\_tfidf(text, number\_of\_sentences=5):

original\_sentences = nltk.sent\_tokenize(text)

if len(original\_sentences) == 0:

return []

# Calculate TF-IDF scores for each sentence

sentence\_scores = calculate\_sentence\_tfidf(original\_sentences)

# Rank sentences based on their TF-IDF score (higher is better)

ranked\_sentences = sorted(((score, sentence) for score, sentence in zip(sentence\_scores, original\_sentences)), reverse=True)

# Return the top N sentences for the summary

summary = [ranked\_sentences[i][1] for i in range(min(number\_of\_sentences, len(ranked\_sentences)))]

return summary