Home.html

<div class="home-container">

  <!-- Left Column: Prompt Section & History -->

  <div class="prompt-section">

    <div class="prompt-card">

      <h2 class="title">Azure AI search</h2>

      <p class="subtitle">AutoMates</p>

      <label class="sr-only" for="promptInput">Your prompt</label>

      <textarea

        id="promptInput"

        class="prompt-input"

        [(ngModel)]="prompt"

        placeholder="e.g., 'cloud architect with Azure skills'"

        (keydown.control.enter)="submitPrompt()"

        (keydown.meta.enter)="submitPrompt()"

      ></textarea>

      <div class="actions">

        <button

          class="clear-btn"

          type="button"

          (click)="clearPrompt()"

          [disabled]="!prompt.length || isSubmitting"

        >

          Clear

        </button>

        <button

          class="primary-btn"

          type="button"

          (click)="submitPrompt()"

          [disabled]="isSubmitDisabled"

        >

          @if (isSubmitting) {

          <span class="spinner"></span>

          } @else {

          <span>Find Candidates</span>

          }

        </button>

      </div>

    </div>

    <!-- Section for previously submitted prompts with expandable summaries -->

    @if (submittedPrompts.length > 0) {

    <div class="submitted-prompts">

      <h3 class="submitted-title">Recent Searches</h3>

      @for (p of submittedPrompts.slice().reverse(); track p.prompt) {

      <div class="prompt-card-small">

        <div class="prompt-history-header" (click)="p.show = !p.show">

          <span

            >"<strong>{{ p.prompt }}</strong

            >" ({{ p.count }})</span

          >

          <span class="toggle-arrow" [class.rotated]="p.show">▼</span>

        </div>

        @if (p.show) {

        <div class="summary-content">

          <p>

            <strong>Summary:</strong> <em>{{ p.summary }}</em>

          </p>

        </div>

        }

      </div>

      }

    </div>

    }

  </div>

  <!-- Right Column: Results Section -->

  <div class="results-section">

    @if (results.length > 0) {

    <div class="results-header">

      <h3>Showing {{ resultCount }} results for:</h3>

      <p class="last-prompt">"{{ lastPrompt }}"</p>

    </div>

    @if (summary) {

    <div class="summary-card">

      <h4>AI Summary</h4>

      <p>{{ summary }}</p>

    </div>

    }

    <!-- Collapsible section for search details -->

    @if (intent || searchRequest) {

    <details class="details-dropdown">

      <summary>Show Search Details</summary>

      @if (intent) {

      <h4>Intent</h4>

      <pre class="code-block">{{ intent | json }}</pre>

      } @if (searchRequest) {

      <h4>Azure Search Request</h4>

      <pre class="code-block">{{ searchRequest | json }}</pre>

      }

    </details>

    }

    <div class="profile-grid">

      @for (profile of results; track profile.recordid; let i = $index) {

      <div class="profile-card" [style.animation-delay]="i \* 100 + 'ms'">

        <div class="profile-header">

          <div class="profile-avatar">{{ profile.Name | slice : 0 : 1 }}</div>

          <div class="profile-info">

            <h4 class="profile-name">{{ profile.Name }}</h4>

            <p class="profile-role">{{ profile.Role }}</p>

          </div>

        </div>

        <div class="profile-body">

          <p class="profile-summary">{{ profile.ProfileSummary }}</p>

          <div class="profile-details">

            <span><strong>Location:</strong> {{ profile.Location }}</span>

            <span><strong>Experience:</strong> {{ profile.ExperienceYears }} years</span>

            <span><strong>Industry:</strong> {{ profile.Industry }}</span>

          </div>

          <div class="profile-skills">

            <strong>Skills:</strong>

            <div class="skills-tags">

              @for (skill of profile.Skills.split(', '); track skill) {

              <span class="skill-tag">{{ skill }}</span>

              }

            </div>

          </div>

        </div>

      </div>

      }

    </div>

    } @else {

    <div class="placeholder">

      <div class="placeholder-icon">📄</div>

      <h3>Your results will appear here</h3>

      <p>Enter a prompt to find candidates.</p>

    </div>

    }

  </div>

</div>

Home.css  
  
:root {

  --background-dark: #121212;

  --background-light: #1e1e1e;

  --primary-color: #3f51b5;

  --primary-hover: #303f9f;

  --text-primary: #ffffff;

  --text-secondary: #b0b0b0;

  --border-color: #333333;

  --card-shadow: 0 10px 30px rgba(0, 0, 0, 0.5);

}

html,

body {

  height: 100%;

  margin: 0;

  font-family: 'Segoe UI', system-ui, sans-serif;

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

  color: var(--text-primary);

}

.home-container {

  display: flex;

  height: 100vh;

  overflow: hidden;

}

/\* Left Column: Prompt Section \*/

.prompt-section {

  width: 35%;

  min-width: 400px;

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

  padding: 2rem;

  display: flex;

  flex-direction: column;

  justify-content: center;

  border-right: 1px solid var(--border-color);

}

.prompt-card {

  animation: fadeIn 0.5s ease-out;

}

.title {

  font-size: 2.5rem;

  font-weight: 700;

  color: var(--primary-color);

  margin: 0;

}

.subtitle {

  font-size: 1.2rem;

  color: var(--text-secondary);

  margin: 0.5rem 0 2rem 0;

}

.prompt-input {

  width: 100%;

  min-height: 150px;

  background-color: #2c2c2c;

  color: var(--text-primary);

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

  border-radius: 8px;

  padding: 1rem;

  font-size: 1rem;

  resize: vertical;

  outline: none;

  transition: border-color 0.2s, box-shadow 0.2s;

}

.prompt-input:focus {

  border-color: var(--primary-color);

  box-shadow: 0 0 0 3px rgba(63, 81, 181, 0.4);

}

.actions {

  display: flex;

  justify-content: flex-end;

  gap: 1rem;

  margin-top: 1rem;

}

.clear-btn,

.primary-btn {

  padding: 0.75rem 1.5rem;

  border: none;

  border-radius: 8px;

  font-size: 1rem;

  font-weight: 600;

  cursor: pointer;

  transition: all 0.2s ease;

}

.clear-btn {

  background-color: transparent;

  color: var(--text-secondary);

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

}

.clear-btn:hover {

  background-color: #333;

  color: var(--text-primary);

}

/\* --- FIX STARTS HERE --- \*/

/\* Styles for the primary button when it is ENABLED \*/

.primary-btn:not(:disabled) {

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

  color: var(--text-primary); /\* Use variable for consistency \*/

  display: flex;

  align-items: center;

  justify-content: center;

}

/\* Hover state for ENABLED button \*/

.primary-btn:not(:disabled):hover {

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

}

/\* Active (on-click) state for ENABLED button \*/

.primary-btn:not(:disabled):active {

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

  transform: translateY(1px); /\* Adds a subtle press effect \*/

}

/\* Styles for the button when it is DISABLED \*/

.primary-btn:disabled {

  background-color: #555;

  color: #a0a0a0; /\* Muted text color for disabled state \*/

  cursor: not-allowed;

  display: flex;

  align-items: center;

  justify-content: center;

}

/\* --- FIX ENDS HERE --- \*/

/\* Right Column: Results Section \*/

.results-section {

  flex-grow: 1;

  padding: 2rem;

  overflow-y: auto;

}

.results-header {

  margin-bottom: 1.5rem;

}

.results-header h3 {

  color: var(--text-secondary);

  margin: 0;

}

.last-prompt {

  font-size: 1.5rem;

  font-style: italic;

  color: var(--text-primary);

  margin: 0.5rem 0 0 0;

}

.summary-card {

  background-color: rgba(63, 81, 181, 0.1);

  border-left: 4px solid var(--primary-color);

  padding: 1.5rem;

  border-radius: 8px;

  margin-bottom: 2rem;

  animation: fadeIn 0.5s ease-out;

}

.summary-card h4 {

  margin: 0 0 0.5rem 0;

  color: var(--primary-color);

}

.profile-grid {

  display: grid;

  grid-template-columns: repeat(auto-fill, minmax(300px, 1fr));

  gap: 1.5rem;

}

/\* Profile Card Design \*/

.profile-card {

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

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

  border-radius: 12px;

  padding: 1.5rem;

  box-shadow: 0 4px 15px rgba(0, 0, 0, 0.3);

  opacity: 0;

  transform: translateY(20px);

  animation: slideInUp 0.5s forwards;

}

.profile-header {

  display: flex;

  align-items: center;

  margin-bottom: 1rem;

}

.profile-avatar {

  width: 50px;

  height: 50px;

  border-radius: 50%;

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

  color: white;

  display: flex;

  align-items: center;

  justify-content: center;

  font-size: 1.5rem;

  font-weight: 600;

  margin-right: 1rem;

}

.profile-name {

  margin: 0;

  font-size: 1.25rem;

}

.profile-role {

  margin: 0;

  color: var(--text-secondary);

}

.profile-summary {

  margin-bottom: 1rem;

  color: var(--text-secondary);

}

.profile-details {

  display: flex;

  flex-wrap: wrap;

  gap: 1rem;

  margin-bottom: 1rem;

  font-size: 0.9rem;

}

.profile-skills {

  margin-top: 1rem;

}

.skills-tags {

  display: flex;

  flex-wrap: wrap;

  gap: 0.5rem;

  margin-top: 0.5rem;

}

.skill-tag {

  background-color: #333;

  color: var(--text-secondary);

  padding: 0.25rem 0.75rem;

  border-radius: 15px;

  font-size: 0.8rem;

}

/\* Placeholder for no results \*/

.placeholder {

  display: flex;

  flex-direction: column;

  align-items: center;

  justify-content: center;

  height: 100%;

  text-align: center;

  color: var(--text-secondary);

}

.placeholder-icon {

  font-size: 4rem;

  margin-bottom: 1rem;

}

/\* Spinner Animation \*/

.spinner {

  width: 20px;

  height: 20px;

  border: 3px solid rgba(255, 255, 255, 0.3);

  border-radius: 50%;

  border-top-color: #fff;

  animation: spin 1s ease-in-out infinite;

}

@keyframes spin {

  to {

    transform: rotate(360deg);

  }

}

@keyframes fadeIn {

  from {

    opacity: 0;

  }

  to {

    opacity: 1;

  }

}

@keyframes slideInUp {

  from {

    opacity: 0;

    transform: translateY(20px);

  }

  to {

    opacity: 1;

    transform: translateY(0);

  }

}

/\* Screen reader helper \*/

.sr-only {

  position: absolute;

  width: 1px;

  height: 1px;

  padding: 0;

  margin: -1px;

  overflow: hidden;

  clip: rect(0, 0, 0, 0);

  white-space: nowrap;

  border: 0;

}

/\* Animation \*/

@keyframes floatIn {

  from {

    opacity: 0;

    transform: translateY(8px) scale(0.99);

  }

  to {

    opacity: 1;

    transform: translateY(0) scale(1);

  }

}

/\* Submitted prompts section \*/

.submitted-prompts {

  margin-top: 16px;

}

.submitted-title {

  margin-bottom: 8px;

  font-weight: 600;

  font-size: 1rem;

  color: #25369b; /\* same as your title color \*/

}

.prompt-card-small {

  background: rgba(141, 148, 179, 0.2); /\* subtle shade, matches dark-theme style \*/

  padding: 10px 14px;

  border-radius: 12px;

  margin-bottom: 8px;

  font-size: 0.95rem;

  color: var(--fg); /\* same text color as original \*/

  word-wrap: break-word;

  box-shadow: inset 0 1px 2px rgba(0, 0, 0, 0.1);

}

/\* IMPORTANT: If you had a light-mode block elsewhere, this keeps us dark \*/

@media (prefers-color-scheme: light) {

  .home-center.dark-theme {

    /\* Stay dark even if system is light \*/

    background: radial-gradient(80rem 50rem at 10% 0%, #152043 0%, transparent 55%),

      radial-gradient(70rem 40rem at 90% 100%, #0d3b66 0%, transparent 55%),

      linear-gradient(180deg, var(--bg) 0%, var(--bg) 100%);

    color: var(--fg);

  }

  .home-center.dark-theme .prompt-card {

    background: var(--card);

    border-color: var(--border);

    box-shadow: 0 20px 60px rgba(0, 0, 0, 0.35), inset 0 1px 0 rgba(255, 255, 255, 0.04);

  }

  .home-center.dark-theme .clear-btn {

    color: var(--fg);

    border-color: var(--border);

  }

  .home-center.dark-theme .hint {

    color: var(--muted);

  }

}

.api-results {

  margin-top: 2rem;

  display: flex;

  flex-wrap: wrap;

  gap: 1rem;

}

.submitted-title {

  width: 100%;

  text-align: center;

  letter-spacing: 0.5rem;

  font-size: 1.5rem;

  text-transform: uppercase;

}

.profile-card {

  background-color: #1e1e1e;

  border: 1px solid #444;

  padding: 1rem;

  margin-bottom: 1rem;

  border-radius: 8px;

  color: #eee;

}

.profile-card h4 {

  margin: 0 0 0.5rem;

  letter-spacing: 0.3rem;

}

.profile-card .role {

  font-weight: normal;

  font-size: 0.9em;

  color: #aaa;

}

.profile-card ul {

  list-style: none;

  padding: 0;

  margin: 0;

}

.profile-card li {

  margin-bottom: 0.5rem;

}

/\* Toggle summary button \*/

.toggle-summary-btn {

  background: none;

  border: none;

  cursor: pointer;

  font-size: 1rem;

  margin-left: 0.5rem;

  color: #00bcd4;

  display: inline-flex;

  align-items: center;

  padding: 0;

}

.toggle-summary-btn span {

  display: inline-block;

  transition: transform 0.3s ease;

  font-size: 1.1rem;

  margin-left: 0.25rem;

}

.toggle-summary-btn span.rotated {

  transform: rotate(180deg);

}

/\* Collapsible summary content \*/

.summary-content {

  margin-top: 0.5rem;

  padding: 0.75rem 1rem;

  background-color: rgba(255, 255, 255, 0.05);

  border-left: 4px solid #00bcd4;

  border-radius: 6px;

  color: var(--fg);

  animation: fadeIn 300ms ease-in-out;

}

/\* Fade-in animation for summary \*/

@keyframes fadeIn {

  from {

    opacity: 0;

    transform: translateY(-4px);

  }

  to {

    opacity: 1;

    transform: translateY(0);

  }

}

Home.ts  
  
import { Component } from '@angular/core';

import { CommonModule } from '@angular/common';

import { FormsModule } from '@angular/forms';

import { SearchService } from '../api/search-search';

@Component({

  selector: 'app-home',

  standalone: true,

  imports: [CommonModule, FormsModule],

  templateUrl: './home.html',

  styleUrl: './home.css',

})

export class Home {

  showSummary: boolean = false;

  prompt: string = '';

  isSubmitting = false;

  lastPrompt: string = '';

  submittedPrompts: { prompt: string; count: number; summary: string; show?: boolean }[] = [];

  results: any[] = [];

  resultCount: number = 0;

  summary: string = '';

  intent: any = null;

  searchRequest: any = null;

  constructor(private searchService: SearchService) {}

  submitPrompt(): void {

    const text = this.prompt.trim();

    if (!text || this.isSubmitting) return;

    this.isSubmitting = true;

    this.results = [];

    this.summary = '';

    this.intent = null;

    this.searchRequest = null;

    this.searchService.searchProfiles(text).subscribe({

      next: (response) => {

        console.log('API response:', response);

        this.lastPrompt = text;

        // Updated to match new backend structure

        this.intent = response.intent;

        this.searchRequest = response.azure\_search\_request;

        this.results = response.results || [];

        this.summary = response.summary || '';

        this.resultCount = this.results.length;

        this.submittedPrompts.push({

          prompt: text,

          count: this.resultCount,

          summary: this.summary,

          show: false,

        });

        this.prompt = '';

        this.isSubmitting = false;

      },

      error: (err) => {

        console.error('API error:', err);

        this.isSubmitting = false;

      },

    });

  }

  clearPrompt(): void {

    if (this.isSubmitting) return;

    this.prompt = '';

  }

  toggleSummary(): void {

    this.showSummary = !this.showSummary;

  }

  get isSubmitDisabled(): boolean {

    return !(this.prompt ?? '').trim() || this.isSubmitting;

  }

}

Voice  
import { environment } from '../../environments/environment';

import { Component, OnInit, signal, computed } from '@angular/core';

import { CommonModule, JsonPipe } from '@angular/common';

import { FormsModule } from '@angular/forms';

import { firstValueFrom } from 'rxjs';

import { Speech } from '../service/speech';

import { Search, SearchResponse } from '../service/search';

type SearchHistoryItem = {

  query: string;

  summary: string;

  results: any[];

  timestamp: Date;

};

@Component({

  selector: 'app-voice',

  imports: [CommonModule, FormsModule, JsonPipe],

  templateUrl: './voice.html',

  styleUrl: './voice.css',

})

export class Voice implements OnInit {

  topK = environment.search?.default\_topk ?? 3;

  sending = signal(false);

  errorMsg = signal('');

  intent = signal<string>('');

  searchRequest = signal<any>(null);

  results = signal<any[]>([]);

  lastSummary = signal('');

  isSpeaking = signal(false);

  micInitializing = signal(false); // New state for initialization

  searchHistory = signal<SearchHistoryItem[]>([]);

  constructor(private speech: Speech, private search: Search) {}

  partial$ = signal('');

  final$ = signal('');

  isPulsing = computed(

    () => this.isListening() && (this.partial$() !== '' || this.final$() !== '')

  );

  ngOnInit() {

    this.partial$ = this.speech.partialText;

    this.final$ = this.speech.finalText;

  }

  isListening() {

    return this.speech.isListening();

  }

  async toggleMic() {

    this.errorMsg.set('');

    // If already listening, the only action is to stop.

    if (this.isListening()) {

      this.speech.stopListening();

      return;

    }

    // Prevent multiple clicks while the mic is starting up.

    if (this.micInitializing()) {

      return;

    }

    this.micInitializing.set(true);

    try {

      this.clearAll();

      await this.speech.startListening();

    } catch (e: any) {

      this.errorMsg.set(e?.message ?? 'Microphone error');

    } finally {

      this.micInitializing.set(false); // Reset state regardless of success or failure

    }

  }

  clearAll() {

    this.speech.resetTranscript();

    this.clearSearchResults();

  }

  clearSearchResults() {

    this.intent.set('');

    this.searchRequest.set(null);

    this.results.set([]);

    this.lastSummary.set('');

    this.errorMsg.set('');

  }

  async sendToSearch() {

    const prompt = this.final$().trim();

    if (!prompt) {

      this.errorMsg.set('Please speak first before searching.');

      return;

    }

    if (this.isListening()) {

      this.speech.stopListening();

    }

    this.sending.set(true);

    this.clearSearchResults();

    try {

      const res: SearchResponse = await firstValueFrom(

        this.search.searchProfiles(prompt, this.topK)

      );

      this.intent.set(res.intent || '');

      this.searchRequest.set(res.azure\_search\_request ?? null);

      this.results.set(res.results ?? []);

      this.lastSummary.set((res.summary ?? '').trim());

      if (this.lastSummary() || this.results().length > 0) {

        this.searchHistory.update((history) => [

          ...history,

          {

            query: prompt,

            summary: this.lastSummary(),

            results: this.results(),

            timestamp: new Date(),

          },

        ]);

      }

    } catch (e: any) {

      this.errorMsg.set(e?.message ?? 'Search API failed');

    } finally {

      this.sending.set(false);

    }

  }

  async speakSummary() {

    if (this.lastSummary()) {

      this.isSpeaking.set(true);

      await this.speech.speak(this.lastSummary());

      this.isSpeaking.set(false);

    }

  }

}

Html  
<div class="jarvis-container">

  <header>

    <h1>AutoMates</h1>

    <p>Voice Assistant</p>

  </header>

  <div class="mic-container">

    <button

      class="mic-button"

      [class.listening]="isListening()"

      [class.initializing]="micInitializing()"

      (click)="toggleMic()"

      [disabled]="micInitializing()"

      aria-label="Toggle Voice Search"

    >

      @if (micInitializing()) {

      <span class="mic-icon loading-spinner"></span>

      } @else {

      <span class="mic-icon">🎙️</span>

      }

    </button>

    <div class="pulse-ring" [class.active]="isPulsing()"></div>

  </div>

  <div class="status">

    @if (micInitializing()) {

    <p class="initializing-text">Initializing microphone...</p>

    } @else if (isListening()) {

    <p class="listening-text">Listening...</p>

    } @else if (isSpeaking()) {

    <p class="speaking-text">Speaking...</p>

    } @else {

    <p>Press the button to start.</p>

    }

  </div>

  <!-- AI Speaking Visualizer -->

  <div class="speaking-visualizer" [class.active]="isSpeaking()">

    <div class="bar"></div>

    <div class="bar"></div>

    <div class="bar"></div>

    <div class="bar"></div>

    <div class="bar"></div>

  </div>

  <div class="transcript-panel">

    <p class="transcript">

      <span class="final">{{ final$() }}</span>

      @if (partial$()) {

      <span class="partial">{{ partial$() }}</span>

      } @if (!final$() && !partial$()) {

      <span class="placeholder">Your command will appear here...</span>

      }

    </p>

  </div>

  <div class="controls-panel">

    <button class="search-button" [disabled]="sending() || !final$()" (click)="sendToSearch()">

      {{ sending() ? 'Searching...' : 'Send to Search' }}

    </button>

  </div>

  @if (errorMsg()) {

  <p class="error-message">{{ errorMsg() }}</p>

  } @if (lastSummary() || results().length > 0) {

  <div class="results-container">

    <h3>Current Result</h3>

    @if (lastSummary()) {

    <div class="summary-panel">

      <div class="summary-header">

        <h4>Summary</h4>

        <button class="speak-button" (click)="speakSummary()" [disabled]="isSpeaking()">

          {{ isSpeaking() ? 'Speaking...' : 'Speak' }}

        </button>

      </div>

      <p>{{ lastSummary() }}</p>

    </div>

    }

    <details class="details-dropdown">

      <summary>Show Records & Search Details</summary>

      @if (intent()) {

      <p><strong>Intent:</strong> {{ intent() }}</p>

      } @if (searchRequest()) {

      <h4>Azure Search Request</h4>

      <pre class="code-block">{{ searchRequest() | json }}</pre>

      } @if (results().length) {

      <h4>Retrieved Records ({{ results().length }})</h4>

      <ol>

        @for (r of results(); track r) {

        <li>

          <pre class="code-block">{{ r | json }}</pre>

        </li>

        }

      </ol>

      }

    </details>

  </div>

  } @if (searchHistory().length > 0) {

  <div class="history-container">

    <h3>Session History</h3>

    @for (item of searchHistory().slice().reverse(); track item.timestamp) {

    <div class="history-item">

      <p><strong>Query:</strong> "{{ item.query }}"</p>

      @if (item.summary) {

      <p><strong>Summary:</strong> {{ item.summary }}</p>

      }

      <details class="details-dropdown">

        <summary>View Retrieved Records ({{ item.results.length }})</summary>

        <ol>

          @for (r of item.results; track r) {

          <li>

            <pre class="code-block">{{ r | json }}</pre>

          </li>

          }

        </ol>

      </details>

    </div>

    }

  </div>

  }

</div>

:host {

  --primary-color: #00bfff; /\* Deep Sky Blue \*/

  --background-color: #1a1a1a;

  --text-color: #e0e0e0;

  --panel-bg-color: #2a2a2a;

  --border-color: #444;

}

.jarvis-container {

  max-width: 800px;

  margin: 2rem auto;

  padding: 2rem;

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

  color: var(--text-color);

  border-radius: 15px;

  box-shadow: 0 10px 30px rgba(0, 0, 0, 0.5);

  font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

}

header {

  text-align: center;

  margin-bottom: 2rem;

}

header h1 {

  color: var(--primary-color);

  margin: 0;

  letter-spacing: 4px;

}

.mic-container {

  position: relative;

  display: flex;

  justify-content: center;

  align-items: center;

  margin-bottom: 1rem;

}

.mic-button {

  width: 100px;

  height: 100px;

  border-radius: 50%;

  background: var(--panel-bg-color);

  border: 3px solid var(--primary-color);

  cursor: pointer;

  display: flex;

  justify-content: center;

  align-items: center;

  transition: all 0.3s ease;

  box-shadow: 0 0 15px rgba(0, 191, 255, 0.3);

  z-index: 2;

}

.mic-button:hover {

  box-shadow: 0 0 25px var(--primary-color);

}

.mic-button.listening {

  background: var(--primary-color);

  box-shadow: 0 0 30px var(--primary-color);

}

.mic-button:disabled,

.mic-button.initializing {

  cursor: wait;

  background-color: #444;

  border-color: #777;

  box-shadow: none;

}

.mic-icon {

  font-size: 3rem;

}

/\* New Spinner Styles \*/

.mic-icon.loading-spinner {

  width: 40px;

  height: 40px;

  border: 5px solid rgba(255, 255, 255, 0.3);

  border-radius: 50%;

  border-top-color: var(--primary-color);

  animation: spin 1s ease-in-out infinite;

}

@keyframes spin {

  to {

    transform: rotate(360deg);

  }

}

.pulse-ring {

  position: absolute;

  width: 100px;

  height: 100px;

  border: 3px solid var(--primary-color);

  border-radius: 50%;

  opacity: 0;

  transform: scale(1);

  z-index: 1;

}

.pulse-ring.active {

  animation: pulse 1.5s infinite;

}

@keyframes pulse {

  0% {

    transform: scale(1);

    opacity: 0.8;

  }

  100% {

    transform: scale(2);

    opacity: 0;

  }

}

.status {

  text-align: center;

  font-size: 1.1rem;

  min-height: 24px;

}

.listening-text {

  color: var(--primary-color);

}

.speaking-text {

  color: #ffdb58;

} /\* Mustard Yellow \*/

.initializing-text {

  color: #aaa;

} /\* New status text color \*/

.speaking-visualizer {

  display: flex;

  align-items: center;

  justify-content: center;

  height: 50px;

  opacity: 0;

  transition: opacity 0.3s ease;

  margin-top: -20px;

  margin-bottom: 1rem;

}

.speaking-visualizer.active {

  opacity: 1;

}

.speaking-visualizer .bar {

  width: 8px;

  height: 5px;

  margin: 0 3px;

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

  border-radius: 5px;

  animation: speak-wave 1.2s infinite ease-in-out;

}

.speaking-visualizer .bar:nth-child(2) {

  animation-delay: 0.2s;

}

.speaking-visualizer .bar:nth-child(3) {

  animation-delay: 0.4s;

}

.speaking-visualizer .bar:nth-child(4) {

  animation-delay: 0.6s;

}

.speaking-visualizer .bar:nth-child(5) {

  animation-delay: 0.8s;

}

@keyframes speak-wave {

  0%,

  100% {

    transform: scaleY(1);

  }

  50% {

    transform: scaleY(5);

  }

}

.transcript-panel {

  background: var(--panel-bg-color);

  padding: 1.5rem;

  border-radius: 10px;

  min-height: 60px;

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

  margin-bottom: 1rem;

}

.transcript .final {

  color: var(--text-color);

  font-weight: bold;

}

.transcript .partial {

  color: #aaa;

}

.transcript .placeholder {

  color: #666;

}

.controls-panel {

  text-align: center;

  margin-bottom: 2rem;

}

.search-button {

  padding: 12px 25px;

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

  border: none;

  color: #fff;

  border-radius: 5px;

  cursor: pointer;

  font-size: 1rem;

  transition: background-color 0.3s;

}

.search-button:disabled {

  background-color: #555;

  cursor: not-allowed;

}

.error-message {

  color: #ff4d4d;

  text-align: center;

}

.results-container,

.history-container {

  margin-top: 2rem;

  padding-top: 1rem;

  border-top: 1px solid var(--border-color);

}

.summary-panel {

  background: rgba(0, 191, 255, 0.1);

  padding: 1rem;

  border-radius: 5px;

  margin-bottom: 1rem;

}

.summary-header {

  display: flex;

  justify-content: space-between;

  align-items: center;

  margin-bottom: 0.5rem;

}

.summary-header h4 {

  margin: 0;

}

.speak-button {

  padding: 8px 15px;

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

  border: none;

  color: #fff;

  border-radius: 5px;

  cursor: pointer;

  font-size: 0.9rem;

  transition: background-color 0.3s;

}

.speak-button:disabled {

  background-color: #555;

  cursor: not-allowed;

}

.details-dropdown {

  background: var(--panel-bg-color);

  border-radius: 5px;

  padding: 0.5rem 1rem;

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

}

.details-dropdown summary {

  cursor: pointer;

  font-weight: bold;

  color: var(--primary-color);

}

.code-block {

  background-color: #111;

  padding: 1rem;

  border-radius: 5px;

  white-space: pre-wrap;

  word-wrap: break-word;

  color: #ccc;

}

.history-item {

  background: var(--panel-bg-color);

  padding: 1rem;

  border-radius: 5px;

  margin-bottom: 1rem;

  border-left: 3px solid var(--primary-color);

}