const express = require("express");

const axios = require("axios");

const bodyParser = require("body-parser");

const app = express();

app.use(bodyParser.json());

// ------------------------

// Cognitive Search Config

// ------------------------

const COG\_SEARCH\_SERVICE = "https://aisearch17.search.windows.net";

const COG\_SEARCH\_KEY = "U6aJUyIlkNssQO6oBoNVDoJoZnZLvPKMr4LQr1cH0oAzSeCEYCMs";

const INDEX\_NAME = "product-index";

const SEARCH\_API\_VERSION = "2025-09-01";

// ------------------------

// AI Foundry / GPT Config

// ------------------------

const OPENAI\_RESOURCE = "https://hemal-mg3addke-swedencentral.openai.azure.com/";

const OPENAI\_KEY = "70DsK0AZNsgtS6zeRdyeO3lgKX9Cci2mhXpuBPv0ezrPaRpYkOmVJQQJ99BIACfhMk5XJ3w3AAAAACOGCBsh";

const GPT\_DEPLOYMENT = "gpt-35-turbo-demo";

const API\_VERSION = "2024-12-01-preview";

// ------------------------

// Endpoint to handle frontend query

// ------------------------

app.post("/api/query", async (req, res) => {

const prompt = req.body.prompt;

try {

// 1️⃣ Query Azure Cognitive Search

const searchResponse = await axios.get(

`${COG\_SEARCH\_SERVICE}/indexes/${INDEX\_NAME}/docs`,

{

params: {

'api-version': SEARCH\_API\_VERSION,

search: prompt,

$top: 10

},

headers: {

"api-key": COG\_SEARCH\_KEY

}

}

);

const searchResults = searchResponse.data.value;

// Prepare text for GPT

let combinedText = "Products found:\n";

searchResults.forEach((item, idx) => {

combinedText += `${idx + 1}. ${item.productName} (${item.category}) - $${item.price}\n`;

});

combinedText += `\nAnswer the user query: ${prompt}`;

// 2️⃣ Call Azure OpenAI GPT

const gptResponse = await axios.post(

`${OPENAI\_RESOURCE}/openai/deployments/${GPT\_DEPLOYMENT}/chat/completions?api-version=${API\_VERSION}`,

{

messages: [{ role: "user", content: combinedText }]

},

{

headers: {

"Content-Type": "application/json",

"api-key": OPENAI\_KEY

}

}

);

const gptText = gptResponse.data.choices[0].message.content;

// 3️⃣ Send JSON to frontend

res.json({

searchResults: searchResults,

gptResponse: gptText

});

} catch (error) {

console.error(error.response ? error.response.data : error.message);

res.status(500).json({ error: "Something went wrong" });

}

});

// ------------------------

// Start Server

// ------------------------

const PORT = 3000;

app.listen(PORT, () => {

console.log(`Backend running at http://localhost:${PORT}`);

});  
  
import { Component } from '@angular/core';

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

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

import { HttpClient, HttpClientModule } from '@angular/common/http';

@Component({

selector: 'app-home',

standalone: true,

imports: [CommonModule, FormsModule, HttpClientModule],

templateUrl: './home.html',

styleUrls: ['./home.css']

})

export class Home {

prompt: string = '';

isSubmitting = false;

submittedPrompts: string[] = [];

backendResponse: string = '';

searchResults: any[] = [];

constructor(private http: HttpClient) {}

submitPrompt(): void {

const text = this.prompt.trim();

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

this.isSubmitting = true;

this.backendResponse = '';

this.searchResults = [];

this.http.post<{ searchResults: any[], gptResponse: string }>('http://localhost:3000/api/query', { prompt: text })

.subscribe({

next: (res) => {

this.submittedPrompts.push(text);

this.searchResults = res.searchResults;

this.backendResponse = res.gptResponse;

this.prompt = '';

this.isSubmitting = false;

},

error: (err) => {

console.error(err);

this.backendResponse = '⚠️ Could not connect to backend';

this.isSubmitting = false;

}

});

}

clearPrompt(): void {

if (this.isSubmitting) return;

this.prompt = '';

}

get isSubmitDisabled(): boolean {

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

}

}  
  
<div class="home-center dark-theme" style="--header-h: 0px;">

<section class="prompt-card" role="region" aria-labelledby="promptTitle">

<h2 class="title" id="promptTitle">Ask anything</h2>

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

<textarea

id="promptInput"

class="prompt-input"

[(ngModel)]="prompt"

placeholder="Type your prompt here…"

(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">

{{ isSubmitting ? 'Submitting…' : 'Submit' }}

</button>

</div>

<!-- Display backend search results -->

<div class="search-results" \*ngIf="searchResults.length">

<h3>Search Results:</h3>

<ul>

<li \*ngFor="let item of searchResults">

{{item.productName}} ({{item.category}}) - ${{item.price}}

</li>

</ul>

</div>

<!-- Display GPT response -->

<div class="gpt-response" \*ngIf="backendResponse">

<h3>GPT Response:</h3>

<p>{{backendResponse}}</p>

</div>

<!-- Submitted prompts -->

<div class="submitted-prompts" \*ngIf="submittedPrompts.length">

<h3>Submitted Prompts:</h3>

<div class="prompt-card-small" \*ngFor="let p of submittedPrompts">

{{ p }}

</div>

</div>

</section>

</div>