```
import React, { useState } from 'react';

// Tailwind CSS est suppose etre disponible dans l'environnement.
// Import des icones Lucide pour une meilleure UI.
// Il n'est pas necessaire d'importer lucide-react explicitement ici car c'est un env de demo.
// Nous utiliserons des SVG inline si besoin pour les icones.

// --- Base de Connaissances des Symptomes et Maladies (Traduit de Prolog) ---
const symptomDefinitions = [
    { symptom: 'fievre', disease: 'grippe' },
    { symptom: 'toux', disease: 'grippe' },
    { symptom: 'courbatures', disease: 'grippe' },
    { symptom: 'maux_de_tete', disease: 'grippe' },
    { symptom: 'fatigue', disease: 'grippe' },

    { symptom: 'toux', disease: 'rhume' },
    { symptom: 'nez_qui_coule', disease: 'rhume' },
    { symptom: 'eternuements', disease: 'rhume' },
    { symptom: 'maux_de_gorge', disease: 'rhume' },

    { symptom: 'maux_de_tete', disease: 'migraine' },
    { symptom: 'sensibilite_lumiere', disease: 'migraine' },
    { symptom: 'nausées', disease: 'migraine' },
    { symptom: 'vision_trouble', disease: 'migraine' },

    { symptom: 'douleur_gorge', disease: 'angine' },
    { symptom: 'difficulte_avaler', disease: 'angine' },
    { symptom: 'fievre', disease: 'angine' },
    { symptom: 'ganglions_gonfles', disease: 'angine' },

    { symptom: 'fatigue', disease: 'anemie' },
    { symptom: 'paleur', disease: 'anemie' },
    { symptom: 'essoufflement', disease: 'anemie' },
    { symptom: 'vertiges', disease: 'anemie' },
    { symptom: 'ongles_cassants', disease: 'anemie' },

    { symptom: 'eruptions_cutanees', disease: 'rougeole' },
    { symptom: 'fievre', disease: 'rougeole' },
    { symptom: 'toux', disease: 'rougeole' },
    { symptom: 'yeux_rouges', disease: 'rougeole' },
    { symptom: 'points_koplik', disease: 'rougeole' },

    { symptom: 'douleur_articulaire', disease: 'arthrite' },
    { symptom: 'gonflement_articulaire', disease: 'arthrite' },
    { symptom: 'raideur_matinale', disease: 'arthrite' },
    { symptom: 'rougeur_articulaire', disease: 'arthrite' },

    { symptom: 'brulures_estomac', disease: 'reflux_gastrique' },
```

```
    { symptom: 'regurgitations_acides', disease: 'reflux_gastrique' },
    { symptom: 'douleur_poitrine_brulante', disease: 'reflux_gastrique' },
    { symptom: 'mauvaise_haleine', disease: 'reflux_gastrique' },

    { symptom: 'douleur_poitrine', disease: 'infarctus' }, // URGENCE MEDICALE
    { symptom: 'essoufflement', disease: 'infarctus' },
    { symptom: 'douleur_bras_gauche', disease: 'infarctus' },
    { symptom: 'sueurs_froides', disease: 'infarctus' },

    { symptom: 'urines_frequentes', disease: 'diabete_type2' },
    { symptom: 'soif_excessive', disease: 'diabete_type2' },
    { symptom: 'fatigue', disease: 'diabete_type2' },
    { symptom: 'vision_floue', disease: 'diabete_type2' },
    { symptom: 'perte_poids_inexpliquee', disease: 'diabete_type2' },

    { symptom: 'congestion_nasale', disease: 'sinusite' },
    { symptom: 'douleur_faciale', disease: 'sinusite' },
    { symptom: 'maux_de_tete', disease: 'sinusite' },
    { symptom: 'ecoulement_nasal_jaune_vert', disease: 'sinusite' },
    { symptom: 'pression_sinusale', disease: 'sinusite' },
];

// --- Base de Connaissances des Traitements (Traduit de Prolog) ---
const treatmentDefinitions = [
    { disease: 'grippe', treatment: 'Repos et hydratation.' },
    { disease: 'grippe', treatment: 'Paracetamol pour la fievre et les douleurs.' },
    { disease: 'grippe', treatment: 'Antiviraux (sur prescription medicale).' },

    { disease: 'rhume', treatment: 'Repos.' },
    { disease: 'rhume', treatment: 'Decongestionnants nasaux.' },
    { disease: 'rhume', treatment: 'Pastilles pour la gorge.' },
    { disease: 'rhume', treatment: 'Analgésiques (paracetamol ou ibuprofen) pour les
douleurs.' },

    { disease: 'migraine', treatment: 'Analgésiques specifiques (triptans).' },
    { disease: 'migraine', treatment: 'Anti-inflammatoires non steroïdiens (AINS).' },
    { disease: 'migraine', treatment: 'Repos dans un environnement calme et sombre.' },

    { disease: 'angine', treatment: 'Antibiotiques (si bacterienne, sur prescription).' },
    { disease: 'angine', treatment: 'Anti-inflammatoires.' },
    { disease: 'angine', treatment: 'Gargarismes.' },
    { disease: 'angine', treatment: 'Hydratation.' },

    { disease: 'anemie', treatment: 'Supplements de fer.' },
    { disease: 'anemie', treatment: 'Alimentation riche en fer.' },
    { disease: 'anemie', treatment: 'Vitamines (B12 si anemie pernicieuse).' },

    { disease: 'rougeole', treatment: 'Repos.' },
```

```
    { disease: 'rougeole', treatment: 'Hydratation.' },
    { disease: 'rougeole', treatment: 'Traitement symptomatique de la fievre et de la toux.' },
    { disease: 'rougeole', treatment: 'Isolement pour eviter la propagation.' },

    { disease: 'arthrite', treatment: 'Anti-inflammatoires non steroïdiens (AINS).' },
    { disease: 'arthrite', treatment: 'Corticosteroides.' },
    { disease: 'arthrite', treatment: 'Physiotherapie.' },
    { disease: 'arthrite', treatment: 'Exercices doux. Gestion de la douleur.' },

    { disease: 'reflux_gastrique', treatment: 'Anti-acides.' },
    { disease: 'reflux_gastrique', treatment: 'Inhibiteurs de la pompe a protons (IPP).' },
    { disease: 'reflux_gastrique', treatment: 'Changements alimentaires (eviter aliments
acides, gras, epices).' },

    { disease: 'infarctus', treatment: 'Appel d\'urgence immediat (15).' },
    { disease: 'infarctus', treatment: 'Aspirine.' },
    { disease: 'infarctus', treatment: 'Nitroglycerine.' },
    { disease: 'infarctus', treatment: 'Intervention medicale urgente (angioplastie, pontage).' },

    { disease: 'diabete_type2', treatment: 'Regime alimentaire equilibre.' },
    { disease: 'diabete_type2', treatment: 'Exercice regulier.' },
    { disease: 'diabete_type2', treatment: 'Medicaments oraux.' },
    { disease: 'diabete_type2', treatment: 'Insuline (si necessaire).' },
    { disease: 'diabete_type2', treatment: 'Surveillance reguliere de la glycemie.' },

    { disease: 'sinusite', treatment: 'Decongestionnants.' },
    { disease: 'sinusite', treatment: 'Lavage nasal au serum physiologique.' },
    { disease: 'sinusite', treatment: 'Antibiotiques (si bacterienne).' },
    { disease: 'sinusite', treatment: 'Analgésiques pour la douleur.' },
];

// --- Composant pour la Saisie des Symptomes ---
function SymptomInput({ setUserSymptoms, setCurrentPage }) {
    const [symptomsText, setSymptomsText] = useState('');

    const handleSubmit = () => {
        // Nettoyer et normaliser les symptomes entres par l'utilisateur
        const cleanedSymptoms = symptomsText
            .toLowerCase()
            .split(',')
            .map(symptom => symptom.trim())
            .filter(symptom => symptom.length > 0);

        setUserSymptoms(cleanedSymptoms);
        setCurrentPage('results');
    };

    // Liste de suggestions de symptomes pour aider l'utilisateur
```

```
  const suggestedSymptoms = [
    'fievre', 'toux', 'maux_de_tete', 'courbatures', 'fatigue',
    'nez_qui_coule', 'eternuements', 'maux_de_gorge', 'sensibilite_lumiere',
    'nausées', 'douleur_gorge', 'difficulte_avaler', 'ganglions_gonfles',
    'paleur', 'essoufflement', 'vertiges', 'ongles_cassants',
    'eruptions_cutanees', 'yeux_rouges', 'douleur_articulaire',
    'gonflement_articulaire', 'raideur_matinale', 'brulures_estomac',
    'regurgitations_acides', 'douleur_poitrine', 'douleur_bras_gauche',
    'urines_frequentes', 'soif_excessive', 'vision_floue',
    'congestion_nasale', 'douleur_faciale'
  ].sort(); // Trier alphabetiquement pour une meilleure lisibilite

  return (
    <div className="flex flex-col items-center justify-center min-h-screen bg-gradient-to-br
from-blue-100 to-purple-100 p-4">
      <div className="bg-white p-8 rounded-xl shadow-2xl w-full max-w-md border-t-4
border-blue-500 transform transition-all duration-500 hover:scale-105">
        <h1 className="text-3xl font-extrabold text-gray-800 mb-6 text-center">
          Muni Pharmacie Intelligente
        </h1>
        <p className="text-gray-600 mb-6 text-center">
          Entrez vos symptomes, separes par des virgules (ex: fievre, toux,
maux_de_tete)
        </p>
        <div className="mb-4">
          <label htmlFor="symptoms" className="block text-gray-700 text-sm font-bold
mb-2">
            Vos Symptômes:
          </label>
          <textarea
            id="symptoms"
            rows="5"
            className="shadow appearance-none border rounded-lg w-full py-3 px-4
text-gray-700 leading-tight focus:outline-none focus:ring-2 focus:ring-blue-400
focus:border-transparent transition duration-300 ease-in-out"
            placeholder="Ex: fievre, toux, courbatures, maux_de_tete"
            value={symptomsText}
            onChange={(e) => setSymptomsText(e.target.value)}
          ></textarea>
        </div>
        <button
          onClick={handleSubmit}
          className="w-full bg-blue-600 hover:bg-blue-700 text-white font-bold py-3 px-4
rounded-lg focus:outline-none focus:shadow-outline transform transition duration-300
ease-in-out hover:scale-105 hover:shadow-lg"
        >
          Diagnostiquer
        </button>
```

```
            <div className="mt-8">
                <h2 className="text-lg font-semibold text-gray-700 mb-3">Suggestions de
Symptômes:</h2>
                <div className="flex flex-wrap gap-2">
                    {suggestedSymptoms.map((symptom, index) => (
                        <span
                            key={index}
                            className="bg-gray-200 text-gray-700 text-xs px-3 py-1 rounded-full
cursor-pointer hover:bg-gray-300 transition duration-200"
                            onClick={() => {
                                const currentSymptoms = symptomsText.split(',').map(s =>
s.trim()).filter(s => s.length > 0);
                                if (!currentSymptoms.includes(symptom)) {
                                    setSymptomsText(currentSymptoms.length > 0 ?
`${symptomsText}, ${symptom}` : symptom);
                                }
                            }}
                        >
                            {symptom.replace(/_/g, ' ')}
                        </span>
                    ))}
                </div>
            </div>
        </div>
        {/* Pied de page */}
        <footer className="mt-8 text-gray-600 text-sm text-center">
            &copy; 2025 Muni Pharmacie Intelligente. Tous droits reserves.
        </footer>
    </div>
  );
}

// --- Composant pour l'Affichage des Resultats ---
function ResultsDisplay({ userSymptoms, setCurrentPage }) {
  const [results, setResults] = useState([]);
  // Nouvelle etats pour la fonctionnalite LLM
  const [llmInfo, setLlmInfo] = useState({}); // { diseaseName: { loading: bool, text: string,
error: string } }

  // Effet pour calculer le diagnostic une fois que les symptomes de l'utilisateur sont
disponibles
  React.useEffect(() => {
    const uniqueDiseases = [...new Set(symptomDefinitions.map(def => def.disease))];
    const diagnosticResults = [];
    const THRESHOLD = 50; // Seuil de pourcentage pour afficher une maladie

    uniqueDiseases.forEach(disease => {
```

```
            const diseaseSymptoms = symptomDefinitions
                .filter(def => def.disease === disease)
                .map(def => def.symptom);

            const totalDiseaseSymptoms = diseaseSymptoms.length;
            if (totalDiseaseSymptoms === 0) return; // Eviter la division par zero

            const matchingSymptoms = userSymptoms.filter(symptom =>
                diseaseSymptoms.includes(symptom)
            );

            const numberOfMatches = matchingSymptoms.length;
            const percentage = (numberOfMatches / totalDiseaseSymptoms) * 100;

            if (percentage >= THRESHOLD) {
                const associatedTreatments = treatmentDefinitions
                    .filter(def => def.disease === disease)
                    .map(def => def.treatment);

                diagnosticResults.push({
                    disease: disease,
                    percentage: parseFloat(percentage.toFixed(2)), // Arrondir a 2 decimales
                    treatments: associatedTreatments,
                });
            }
        });

        // Trier les resultats par pourcentage decroissant
        diagnosticResults.sort((a, b) => b.percentage - a.percentage);
        setResults(diagnosticResults);
    }, [userSymptoms]); // Re-executer quand userSymptoms change

    // Fonction pour appeler le LLM
    const fetchLlmExplanation = async (diseaseName) => {
        setLlmInfo(prev => ({ ...prev, [diseaseName]: { loading: true, text: '', error: '' } }));
        const prompt = `Donne une explication simple de la maladie
'${diseaseName.replace(/_/g, ' ')}' et des conseils de sante generale pour la prevenir ou la
gerer. Ne donne pas de conseils medicaux specifiques, juste des informations generales et
preventives, et formate la reponse en paragraphes clairs.`;

        try {
            let chatHistory = [];
            chatHistory.push({ role: "user", parts: [{ text: prompt }] });
            const payload = { contents: chatHistory };
            const apiKey = ""; // Laisser vide, Canvas fournira la cle API en runtime
            const apiUrl =
`https://generativelanguage.googleapis.com/v1beta/models/gemini-2.0-flash:generateConten
t?key=${apiKey}`;
```

```
        const response = await fetch(apiUrl, {
            method: 'POST',
            headers: { 'Content-Type': 'application/json' },
            body: JSON.stringify(payload)
        });
        const result = await response.json();

        if (result.candidates && result.candidates.length > 0 &&
            result.candidates[0].content && result.candidates[0].content.parts &&
            result.candidates[0].content.parts.length > 0) {
            const text = result.candidates[0].content.parts[0].text;
            setLlmInfo(prev => ({ ...prev, [diseaseName]: { loading: false, text: text, error: '' }
}));
        } else {
            setLlmInfo(prev => ({ ...prev, [diseaseName]: { loading: false, text: '', error:
'Reponse LLM inattendue.' } }));
        }
    } catch (error) {
        console.error("Erreur lors de l'appel LLM:", error);
        setLlmInfo(prev => ({ ...prev, [diseaseName]: { loading: false, text: '', error: 'Erreur
reseau ou du serveur.' } }));
    }
};

return (
    <div className="flex flex-col items-center min-h-screen bg-gradient-to-br
from-blue-100 to-purple-100 p-4">
        <div className="bg-white p-8 rounded-xl shadow-2xl w-full max-w-2xl border-t-4
border-blue-500 mt-8">
            <h1 className="text-3xl font-extrabold text-gray-800 mb-6 text-center">
                Resultats du Diagnostic
            </h1>

            <p className="text-gray-700 mb-4 text-center">
                Vos symptomes saisis: <span
className="font-semibold">{userSymptoms.join(', ') || 'Aucun'}</span>
            </p>

            {results.length > 0 ? (
                results.map((result, index) => (
                    <div key={index} className="mb-8 p-6 bg-blue-50 rounded-lg shadow-md
border border-blue-200">
                        <h2 className="text-2xl font-bold text-blue-800 mb-3 capitalize">
                            {result.disease.replace(/_/g, ' ')} ({result.percentage}%)
                        </h2>
                        <h3 className="text-lg font-semibold text-gray-700 mb-2">Traitements
suggérés :</h3>
```

```
                    {result.treatments.length > 0 ? (
                        <ul className="list-disc list-inside text-gray-600 space-y-1">
                            {result.treatments.map((treatment, tIndex) => (
                                <li key={tIndex}>{treatment}</li>
                            ))}
                        </ul>
                    ) : (
                        <p className="text-gray-600 italic">Aucun traitement specifique
enregistre pour cette maladie.</p>
                    )}

                    {/* Bouton pour appeler le LLM */}
                    <button
                        onClick={() => fetchLlmExplanation(result.disease)}
                        className="mt-4 bg-purple-600 hover:bg-purple-700 text-white
font-bold py-2 px-4 rounded-lg focus:outline-none focus:shadow-outline transform transition
duration-300 ease-in-out hover:scale-105 hover:shadow-lg flex items-center justify-center"
                        disabled={llmInfo[result.disease]?.loading}
                    >
                        {llmInfo[result.disease]?.loading ? (
                            <svg className="animate-spin -ml-1 mr-3 h-5 w-5 text-white"
xmlns="http://www.w3.org/2000/svg" fill="none" viewBox="0 0 24 24">
                                <circle className="opacity-25" cx="12" cy="12" r="10"
stroke="currentColor" strokeWidth="4"></circle>
                                <path className="opacity-75" fill="currentColor" d="M4 12a8 8 0
018-8V0C5.373 0 0 5.373 0 12h4zm2 5.291A7.962 7.962 0 014 12H0c0 3.042 1.135 5.824
3 7.938l3-2.647z"></path>
                            </svg>
                        ) : (
                            <>✨ En savoir plus sur la maladie et des conseils</>
                        )}
                    </button>

                    {/* Affichage des informations du LLM */}
                    {llmInfo[result.disease]?.text && (
                        <div className="mt-4 p-4 bg-gray-50 rounded-md border
border-gray-200">
                            <h4 className="text-lg font-semibold text-gray-800
mb-2">Informations supplémentaires ✨</h4>
                            <p className="text-gray-700
whitespace-pre-wrap">{llmInfo[result.disease].text}</p>
                        </div>
                    )}
                    {llmInfo[result.disease]?.error && (
                        <div className="mt-4 p-4 bg-red-100 text-red-700 rounded-md border
border-red-300">
                            <p className="font-bold">Erreur:</p>
                            <p>{llmInfo[result.disease].error}</p>
```

```
                        </div>
                    )}
                </div>
            ))
        ) : (
            <div className="bg-yellow-50 border-l-4 border-yellow-400 text-yellow-700 p-4 rounded-md" role="alert">
                <p className="font-bold">Information:</p>
                <p>Aucune maladie significative (avec au moins 50% de correspondance) n'a été identifiée pour les symptômes saisis.</p>
            </div>
        )}

        <div className="mt-8 p-4 bg-red-50 border-l-4 border-red-400 text-red-700 rounded-md" role="alert">
            <p className="font-bold">Avertissement Médical Important:</p>
            <p className="text-sm mt-2">
                Cette application est un outil informatif et ne remplace en aucun cas un avis, un diagnostic ou un traitement médical professionnel. Consultez toujours un professionnel de la santé qualifié pour tout problème de santé. L'auto-diagnostic et l'auto-médication peuvent être dangereux.
            </p>
        </div>

        <button
            onClick={() => setCurrentPage('input')}
            className="mt-8 w-full bg-gray-600 hover:bg-gray-700 text-white font-bold py-3 px-4 rounded-lg focus:outline-none focus:shadow-outline transform transition duration-300 ease-in-out hover:scale-105 hover:shadow-lg"
        >
            Retour à la Saisie des Symptômes
        </button>
    </div>
    {/* Pied de page */}
    <footer className="mt-8 text-gray-600 text-sm text-center">
        &copy; 2025 Muni Pharmacie Intelligente. Tous droits reserves.
    </footer>
  </div>
);
}

// --- Composant Principal de l'Application ---
export default function App() {
  const [currentPage, setCurrentPage] = useState('input'); // 'input' ou 'results'
  const [userSymptoms, setUserSymptoms] = useState([]);

  switch (currentPage) {
    case 'input':
```

```
            return <SymptomInput setUserSymptoms={setUserSymptoms}
setCurrentPage={setCurrentPage} />;
        case 'results':
            return <ResultsDisplay userSymptoms={userSymptoms}
setCurrentPage={setCurrentPage} />;
        default:
            return <SymptomInput setUserSymptoms={setUserSymptoms}
setCurrentPage={setCurrentPage} />;
    }
}
```