```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Muni Pharmacie Intelligente</title>
  <!-- Chargement de Tailwind CSS pour un style moderne et responsive -->
  <script src="https://cdn.tailwindcss.com"></script>
  <link
href="https://fonts.googleapis.com/css2?family=Inter:wght@400;600;800&display=swap"
rel="stylesheet">
  <style>
    body {
      font-family: 'Inter', sans-serif;
      transition: background 0.3s ease;
    }
    /* Style des cartes de resultat */
    .result-card {
      box-shadow: 0 10px 15px -3px rgba(0, 0, 0, 0.1), 0 4px 6px -2px rgba(0, 0, 0, 0.05);
      transition: transform 0.3s ease, box-shadow 0.3s ease;
    }
    .result-card:hover {
      transform: translateY(-3px);
      box-shadow: 0 20px 25px -5px rgba(0, 0, 0, 0.1), 0 10px 10px -5px rgba(0, 0, 0,
0.04);
    }
    /* Animation de chargement */
    .loader {
      border-top-color: #3b82f6;
      -webkit-animation: spinner 1.5s linear infinite;
      animation: spinner 1.5s linear infinite;
    }
    @-webkit-keyframes spinner {
      0% { -webkit-transform: rotate(0deg); }
      100% { -webkit-transform: rotate(360deg); }
    }
    @keyframes spinner {
      0% { transform: rotate(0deg); }
      100% { transform: rotate(360deg); }
    }
  </style>
</head>
<body class="bg-gray-50">

  <!-- Conteneur principal de l'application -->
  <div id="app" class="min-h-screen flex flex-col items-center p-4 sm:p-8">

    <!-- Conteneur pour le contenu dynamique -->
```

```
        <div id="content" class="w-full max-w-4xl bg-white rounded-xl shadow-2xl p-6 sm:p-10
mt-8 mb-8 border-t-8 border-blue-600">
            <!-- Le contenu sera genere ici par JavaScript -->
        </div>

    </div>

    <script>
        // --- 1. Base de Connaissances (Adaptee 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' }, { 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' },
      ];

      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.' },
    ];

    // --- 2. Variables d'Etat Globales et Utilitaire ---

    let userSymptoms = [];
    let diagnosticResults = [];
    let llmCache = {}; // Cache pour stocker les reponses LLM

    const allSymptoms = [...new Set(symptomDefinitions.map(def => def.symptom))];
    const contentDiv = document.getElementById('content');

    // Formatte un nom de maladie ou de symptome pour l'affichage (ex: 'maux_de_tete' ->
'Maux de Tête')
    const formatName = (name) => {
        return name.replace(/_/g, ' ').split(' ').map(word => word.charAt(0).toUpperCase() +
word.slice(1)).join(' ');
    };
```

```
};

// Fonction utilitaire pour creer des elements HTML
const createElement = (tag, classes, innerHTML = '', id = '') => {
   const el = document.createElement(tag);
   if (classes) el.className = classes;
   if (innerHTML) el.innerHTML = innerHTML;
   if (id) el.id = id;
   return el;
};

// --- 3. Logique de Diagnostic (Fonctionnement similaire a Prolog) ---

const runDiagnosis = (symptoms) => {
   const uniqueDiseases = [...new Set(symptomDefinitions.map(def => def.disease))];
   const results = [];
   const THRESHOLD = 50;

   uniqueDiseases.forEach(disease => {
      const diseaseSymptoms = symptomDefinitions
         .filter(def => def.disease === disease)
         .map(def => def.symptom);

      const totalDiseaseSymptoms = diseaseSymptoms.length;
      if (totalDiseaseSymptoms === 0) return;

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

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

      if (percentage >= THRESHOLD) {
         results.push({
            disease: disease,
            percentage: parseFloat(percentage.toFixed(2)),
            treatments: treatmentDefinitions
               .filter(def => def.disease === disease)
               .map(def => def.treatment),
         });
      }
   });

   results.sort((a, b) => b.percentage - a.percentage);
   diagnosticResults = results;
};


// --- 4. Integration LLM (Gemini API) ---
```

```
        const fetchLlmExplanation = async (diseaseName) => {
            const llmInfoDiv = document.getElementById(`llm-info-${diseaseName}`);
            if (!llmInfoDiv) return;

            // Afficher le cache si disponible
            if (llmCache[diseaseName]) {
                llmInfoDiv.innerHTML = `
                    <h4 class="text-lg font-semibold text-gray-800 mb-2">Informations
supplémentaires ✨</h4>
                    <p class="text-gray-700 whitespace-pre-wrap">${llmCache[diseaseName]}</p>
                `;
                return;
            }

            // Afficher l'indicateur de chargement
            llmInfoDiv.innerHTML = `
                <div class="flex items-center justify-center p-4 bg-gray-100 rounded-lg">
                    <div class="loader ease-linear rounded-full border-4 border-t-4 border-gray-200
h-6 w-6 mr-3"></div>
                    <span class="text-blue-600 font-medium">Recherche d'informations
Gemini...</span>
                </div>
            `;

            const prompt = `Agis comme un conseiller en sante qui donne des informations
generales. Donne une explication simple de la maladie '${formatName(diseaseName)}' 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 utilise des emojis.
Formate la reponse en paragraphes clairs.`;

            const apiKey = "";
            const apiUrl =
`https://generativelanguage.googleapis.com/v1beta/models/gemini-2.5-flash-preview-05-20:
generateContent?key=${apiKey}`;
            const payload = {
                contents: [{ parts: [{ text: prompt }] }],
                systemInstruction: { parts: [{ text: "Tu es un conseiller en sante amical et informatif.
Tes reponses doivent etre en francais." }] },
                tools: [{ "google_search": {} }],
            };

            let responseText = '';
            let errorOccurred = false;
            let attempt = 0;
            const maxAttempts = 3;

            while (attempt < maxAttempts) {
```

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

                if (!response.ok) throw new Error(`HTTP error! status: ${response.status}`);

                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) {
                    responseText = result.candidates[0].content.parts[0].text;
                    llmCache[diseaseName] = responseText; // Mettre en cache la reponse
                    break; // Succes, sortir de la boucle
                } else {
                    throw new Error('Réponse LLM inattendue.');
                }

            } catch (error) {
                console.error(`Attempt ${attempt + 1} failed:`, error);
                errorOccurred = true;
                attempt++;
                if (attempt < maxAttempts) {
                    await new Promise(resolve => setTimeout(resolve, 2 ** attempt * 1000)); //
Exponential backoff
                } else {
                    responseText = 'Impossible d\'obtenir des informations supplementaires pour
le moment. Veuillez reessayer.';
                }
            }
        }

        // Afficher le resultat
        llmInfoDiv.innerHTML = `
            <h4 class="text-lg font-semibold text-gray-800 mb-2">Informations
supplémentaires ✨</h4>
            <p class="text-gray-700 whitespace-pre-wrap">${responseText}</p>
        `;
    };

    // --- 5. Navigation et Rendu des Pages ---

    // Rendu de la page de saisie des symptomes (Page 1)
    const renderInputPage = () => {
        contentDiv.innerHTML = '';
        const title = createElement('h1', 'text-4xl font-extrabold text-gray-900 mb-2
text-center', 'Bienvenue à la Pharmacie Intelligente 💊');
        const subtitle = createElement('p', 'text-xl text-gray-600 mb-8 text-center',
'Sélectionnez tous les symptômes que vous ressentez pour un diagnostic initial.');
        const form = createElement('form', 'space-y-6', '', 'symptom-form');

        // Mise en page de la grille pour les checkboxs
        const symptomsGrid = createElement('div', 'grid grid-cols-2 sm:grid-cols-3
md:grid-cols-4 gap-4');

        allSymptoms.forEach(symptom => {
          const symptomId = `symptom-${symptom}`;
          const div = createElement('div', 'flex items-center bg-blue-50 p-3 rounded-lg
border border-blue-200 hover:bg-blue-100 transition duration-150 ease-in-out
cursor-pointer');

          const checkbox = createElement('input', 'h-5 w-5 text-blue-600 border-gray-300
rounded focus:ring-blue-500', '', symptomId);
          checkbox.type = 'checkbox';
          checkbox.name = 'symptom';
          checkbox.value = symptom;

          const label = createElement('label', 'ml-3 text-sm font-medium text-gray-700
select-none flex-grow', formatName(symptom));
          label.htmlFor = symptomId;

          // Toggle l'etat de la checkbox en cliquant sur la div
          div.onclick = () => { checkbox.checked = !checkbox.checked; };

          div.appendChild(checkbox);
          div.appendChild(label);
          symptomsGrid.appendChild(div);
        });

        const submitButton = createElement('button', 'w-full py-3 px-4 bg-blue-600
hover:bg-blue-700 text-white font-bold rounded-lg transition duration-200 ease-in-out
transform hover:scale-[1.01] shadow-lg focus:outline-none focus:ring-4 focus:ring-blue-500
focus:ring-opacity-50', 'Soumettre les Symptômes et Diagnostiquer');
        submitButton.type = 'submit';

        form.appendChild(symptomsGrid);
        form.appendChild(submitButton);

        form.onsubmit = (e) => {
          e.preventDefault();
          userSymptoms =
Array.from(form.querySelectorAll('input[name="symptom"]:checked')).map(cb => cb.value);
```

```
        if (userSymptoms.length === 0) {
            alert('Veuillez sélectionner au moins un symptôme.');
            return;
        }

        runDiagnosis(userSymptoms);
        renderResultsPage();
    };

    contentDiv.appendChild(title);
    contentDiv.appendChild(subtitle);
    contentDiv.appendChild(form);
};

// Rendu de la page des resultats (Maladies) (Page 2)
const renderResultsPage = () => {
    contentDiv.innerHTML = '';

    const title = createElement('h1', 'text-4xl font-extrabold text-gray-900 mb-2
text-center', 'Résultats du Diagnostic');

    const backButton = createElement('button', 'w-full py-3 px-4 bg-gray-500
hover:bg-gray-600 text-white font-bold rounded-lg transition duration-200 ease-in-out mb-6',
'← Recommencer la Saisie des Symptômes');
    backButton.onclick = renderInputPage;

    contentDiv.appendChild(title);
    contentDiv.appendChild(backButton);

    // Avertissement Medical
    const alertDiv = createElement('div', 'p-4 mb-6 bg-red-100 border-l-4 border-red-500
text-red-700 rounded-lg', `
        <p class="font-bold">⚠️ Avertissement Médical Important :</p>
        <p class="text-sm mt-1">Ce système est un outil d'information uniquement et ne
remplace pas un avis ou diagnostic médical professionnel. Consultez toujours un
professionnel de la santé.</p>
    `);
    contentDiv.appendChild(alertDiv);

    if (diagnosticResults.length === 0) {
        const noResult = createElement('div', 'p-6 bg-yellow-100 text-yellow-800
rounded-lg text-center font-semibold', 'Aucune maladie significative (>= 50% de
correspondance) n\'a été trouvée pour les symptômes sélectionnés.');
        contentDiv.appendChild(noResult);
        return;
    }
```

```
            const resultsContainer = createElement('div', 'space-y-6');

            diagnosticResults.forEach(result => {
               const card = createElement('div', 'result-card p-6 bg-white rounded-xl border
border-gray-200 shadow-md');

               const header = createElement('div', 'flex items-center justify-between mb-3');
               const diseaseName = createElement('h2', 'text-2xl font-bold text-blue-700
capitalize', formatName(result.disease));
               const percentageBadge = createElement('span', 'px-3 py-1 text-sm font-semibold
rounded-full text-white bg-green-500', `${result.percentage}%`);

               header.appendChild(diseaseName);
               header.appendChild(percentageBadge);
               card.appendChild(header);

               const treatmentList = createElement('ul', 'list-disc pl-5 text-gray-700 mt-3
space-y-1');
               result.treatments.forEach(t => {
                  const li = createElement('li', '', t);
                  treatmentList.appendChild(li);
               });

               const treatmentSection = createElement('div', 'mt-4 border-t pt-4');
               const treatmentTitle = createElement('h3', 'text-lg font-semibold text-gray-800
mb-2', 'Traitements Suggérés (Base de connaissances) :');
               treatmentSection.appendChild(treatmentTitle);
               treatmentSection.appendChild(treatmentList);
               card.appendChild(treatmentSection);


               // Bouton LLM
               const llmButton = createElement('button', 'mt-4 w-full py-2 px-4 bg-purple-600
hover:bg-purple-700 text-white font-bold rounded-lg transition duration-200 ease-in-out
transform hover:scale-[1.01] shadow-md focus:outline-none focus:ring-4
focus:ring-purple-500 focus:ring-opacity-50 flex items-center justify-center text-sm', '✨
Obtenir plus d\'informations et conseils');
               llmButton.onclick = () => fetchLlmExplanation(result.disease);
               card.appendChild(llmButton);

               // Conteneur pour la reponse LLM
               const llmInfoDiv = createElement('div', 'mt-4 p-0 bg-gray-50 rounded-md border
border-gray-200 hidden', '', `llm-info-${result.disease}`);
               card.appendChild(llmInfoDiv);

               llmButton.onclick = () => {
                  llmInfoDiv.classList.toggle('hidden');
                  if (!llmCache[result.disease] && llmInfoDiv.style.display !== 'none') {
                    fetchLlmExplanation(result.disease);
                }
            };

            resultsContainer.appendChild(card);
        });

        contentDiv.appendChild(resultsContainer);
    };

    // --- 6. Initialisation de l'Application ---

    window.onload = renderInputPage;

  </script>
</body>
</html>
```