```javascript
// 🔗 Real GPT-4 AI Connection
function askSupplAi(prompt) {
    fetch("/.netlify/functions/supplai-gpt", {
      method: "POST",
      headers: { "Content-Type": "application/json" },
      body: JSON.stringify({ message: prompt })
    })
    .then(res => res.json())
    .then(data => {
      const assistantBox = document.getElementById('supplai-assistant');
      assistantBox.innerHTML = `<strong>SupplAi:</strong> ${data.reply}`;
    })
    .catch(err => {
      console.error("GPT fetch failed:", err);
    });
  }

  document.addEventListener('DOMContentLoaded', function () {
    console.log('✅ SupplAi script loaded');

    const itemsDatabase = [
      { name: "Blue Chair", category: "Furniture", location: "Toronto" },
      { name: "Red Table", category: "Furniture", location: "Toronto" },
      { name: "Green Sofa", category: "Furniture", location: "Ottawa" },
      { name: "Wooden Desk", category: "Furniture", location: "Montreal" },
      { name: "Bookshelf", category: "Furniture", location: "Vancouver" },
      { name: "Electric Drill", category: "Tools", location: "Toronto" },
      { name: "Toolbox", category: "Tools", location: "Montreal" },
      { name: "Screwdriver Set", category: "Tools", location: "Ottawa" },
      { name: "Socket Wrench", category: "Tools", location: "Calgary" },
      { name: "Hammer", category: "Tools", location: "Toronto" }
    ];

    const cityCoords = {
      Toronto: [43.65107, -79.347015],
      Ottawa: [45.4215, -75.6972],
      Montreal: [45.5017, -73.5673],
      Vancouver: [49.2827, -123.1207],
      Calgary: [51.0447, -114.0719],
      Edmonton: [53.5461, -113.4938],
      Halifax: [44.6488, -63.5752]
    };

    const map = L.map('map').setView([43.65107, -79.347015], 4);
    L.tileLayer('https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png', {
      attribution: '&copy; OpenStreetMap contributors'
    }).addTo(map);

    if (navigator.geolocation) {
      navigator.geolocation.getCurrentPosition(
        (position) => {
          const userLat = position.coords.latitude;
          const userLng = position.coords.longitude;
          map.setView([userLat, userLng], 10);
          L.marker([userLat, userLng])
            .addTo(map)
            .bindPopup("📍 You are here")
            .openPopup();
        },
        (error) => {
          console.warn("Geolocation error:", error.message);
        }
      );
    }

    function updateMapMarkers(filteredItems) {
      map.eachLayer(layer => {
        if (layer instanceof L.Marker) map.removeLayer(layer);
      });
      filteredItems.forEach(item => {
        const coords = cityCoords[item.location];
        if (coords) {
          L.marker(coords).addTo(map).bindPopup(`${item.name}<br>
<b>${item.location}</b>`);
        }
      });
    }

    const searchInput = document.getElementById('searchInput');
    const suggestionsSection = document.getElementById('suggestions');
    const voiceButton = document.getElementById('voiceSearchButton');
    const randomButton = document.getElementById('randomSuggestButton');
    const categorySelect = document.getElementById('categoryFilter');
    const locationInput = document.getElementById('locationInput');

    function smartMatch(query, items) {
      const keywords = query.toLowerCase().split(/\s+/);
      return items.filter(item => {
        const name = item.name.toLowerCase();
        const category = item.category.toLowerCase();
        return keywords.some(k => name.includes(k) || category.includes(k));
      });
    }

    searchInput.addEventListener('input', function () {
      const query = searchInput.value.toLowerCase();
      askSupplAi(query);
      const selectedCategory = categorySelect ? categorySelect.value : 'All';
      const userLocation = locationInput ? locationInput.value.trim().toLowerCase() : '';
      suggestionsSection.innerHTML = '';

      if (query.length > 0) {
        const filtered = smartMatch(query, itemsDatabase).filter(item =>
          (selectedCategory === 'All' || item.category === selectedCategory) &&
          (userLocation === '' || item.location.toLowerCase() === userLocation)
        );

        if (filtered.length > 0) {
          filtered.forEach(item => {
            const div = document.createElement('div');
            div.classList.add('suggestion-item');

            const textSpan = document.createElement('span');
            textSpan.textContent = `${item.name} (${item.location})`;

            const heart = document.createElement('span');
            heart.textContent = '🤍';
            heart.style.cursor = 'pointer';
            heart.style.marginLeft = '10px';

            const favorites = JSON.parse(localStorage.getItem('favorites') || '[]');
            if (favorites.includes(item.name)) {
              heart.textContent = '❤️';
            }

            heart.addEventListener('click', (e) => {
              e.stopPropagation();
              let favs = JSON.parse(localStorage.getItem('favorites') || '[]');
              if (favs.includes(item.name)) {
                favs = favs.filter(f => f !== item.name);
                heart.textContent = '🤍';
              } else {
                favs.push(item.name);
                heart.textContent = '❤️';
              }
              localStorage.setItem('favorites', JSON.stringify(favs));
            });

            div.appendChild(textSpan);
            div.appendChild(heart);
            suggestionsSection.appendChild(div);
          });
          updateMapMarkers(filtered);
        } else {
          suggestionsSection.innerHTML = '<div>No matches found.</div>';
          updateMapMarkers([]);
        }
      }
    });

    if ('webkitSpeechRecognition' in window) {
      const recognition = new webkitSpeechRecognition();
      recognition.lang = 'en-US';
      recognition.interimResults = false;
      recognition.maxAlternatives = 1;

      voiceButton.addEventListener('click', () => {
        recognition.start();
        voiceButton.textContent = '🎙️ Listening...';
      });

      recognition.addEventListener('result', (event) => {
        const speechResult = event.results[0][0].transcript;
        searchInput.value = speechResult;
        searchInput.dispatchEvent(new Event('input'));
        voiceButton.textContent = '🎙️ Voice Search';
      });

      recognition.addEventListener('end', () => {
        voiceButton.textContent = '🎙️ Voice Search';
      });
    } else {
      voiceButton.disabled = true;
      voiceButton.textContent = '🎙️ Not Supported';
    }

    if (randomButton) {
      randomButton.addEventListener('click', () => {
        const randomItem = itemsDatabase[Math.floor(Math.random() *
itemsDatabase.length)];
        searchInput.value = randomItem.name;
        searchInput.dispatchEvent(new Event('input'));
      });
    }

    suggestionsSection.addEventListener('click', (e) => {
      if (e.target.classList.contains('suggestion-item')) {
        const selectedItem = e.target.textContent;
        searchInput.value = selectedItem;
        searchInput.dispatchEvent(new Event('input'));
```

```javascript
      }
    });
  });

```