```
import React, { useState, useEffect } from 'react';
import { Search, Mail, Globe, ChevronDown, X, Menu, Settings,
CheckCircle, AlertTriangle, Zap, Weight, Ruler, ChevronsRight,
Calculator } from 'lucide-react';

// Expanded and more detailed machine data based on the provided
catalog
const machineData = [
  {
    "machine_id": "KMK-WR-SERIES", "name": "WR Weigher",
"description": "Compact devices for weighing and portioning
vegetables. Equipped with a user-friendly Mitsubishi touch screen.",
"vegetable_type": ["potato", "onion"], "function": "Weighing",
"category": "Weighing & Packing",
    "models": [
      { "model_id": "WR-50-MAXI", "input_capacity_kg_h": 9000,
"power_kw": 2, "power_supply": "400V", "dimensions_m": {"length": 3.6,
"width": 1.2, "height": 2.8}, "net_weight_kg": 480 },
      { "model_id": "WR-30", "input_capacity_kg_h": 5500, "power_kw":
2, "power_supply": "400V", "dimensions_m": {"length": 3.3, "width":
1.2, "height": 2.5}, "net_weight_kg": 440 }
    ], "image_url":
"https://placehold.co/600x400/e85d04/ffffff?text=WR+Weigher"
  },
  {
    "machine_id": "KMK-WK-SERIES", "name": "WK Multi-head Weigher",
"description": "Professional multi-head weighers for high efficiency
and accuracy.", "vegetable_type": ["potato", "onion"], "function":
"Weighing", "category": "Weighing & Packing",
    "models": [
      { "model_id": "WK-12", "input_capacity_kg_h": 16000, "power_kw":
3.5, "power_supply": "400V", "dimensions_m": {"length": 2.5, "width":
2.5, "height": 3.0}, "net_weight_kg": 1500 },
      { "model_id": "WK-09", "input_capacity_kg_h": 13000, "power_kw":
3.0, "power_supply": "400V", "dimensions_m": {"length": 2.5, "width":
2.5, "height": 2.7}, "net_weight_kg": 1200 }
    ], "image_url":
"https://placehold.co/600x400/dc2f02/ffffff?text=WK+Weigher"
  },
  {
    "machine_id": "KMK-OS-TOPPER", "name": "OS Onion Topper",
"description": "A stationary device for topping onions using vibrating
screens and high-speed rotating knives.", "vegetable_type": ["onion"],
"function": "Topping", "category": "Cleaning & Washing",
    "models": [
      { "model_id": "OS-1", "input_capacity_kg_h": 4000, "power_kw":
4.0, "power_supply": "400V", "dimensions_m": {"length": 2.0, "width":
1.5, "height": 1.8}, "net_weight_kg": 600 },
```

```
      { "model_id": "OS-2", "input_capacity_kg_h": 8000, "power_kw":
6.0, "power_supply": "400V", "dimensions_m": {"length": 2.5, "width":
1.8, "height": 2.0}, "net_weight_kg": 850 }
    ], "image_url":
"https://placehold.co/600x400/d00000/ffffff?text=Onion+Topper"
  },
  {
    "machine_id": "KMK-S-BRUSH-CLEANER", "name": "S-series Brush
Cleaner", "description": "Designed for dry cleaning of potatoes and
onions using multiple wavy brushes.", "vegetable_type": ["potato",
"onion"], "function": "Cleaning", "category": "Cleaning & Washing",
    "models": [
      { "model_id": "S-10", "input_capacity_kg_h": 10000, "power_kw":
2.2, "power_supply": "400V", "dimensions_m": {"length": 3.0, "width":
1.5, "height": 2.0}, "net_weight_kg": 700 }
    ], "image_url":
"https://placehold.co/600x400/9d0208/ffffff?text=Brush+Cleaner"
  },
  {
    "machine_id": "KMK-MD-WASHER", "name": "MD-series Drum Washer",
"description": "A series of drum washers with a perforated,
acid-resistant stainless steel drum for washing vegetables.",
"vegetable_type": ["potato"], "function": "Washing", "category":
"Cleaning & Washing",
    "models": [
      { "model_id": "MD-1", "input_capacity_kg_h": 12000, "power_kw":
3.0, "power_supply": "400V", "dimensions_m": {"length": 4.0, "width":
2.0, "height": 2.5}, "net_weight_kg": 1100 }
    ], "image_url":
"https://placehold.co/600x400/6a040f/ffffff?text=Drum+Washer"
  },
  {
    "machine_id": "KMK-SO-SIZER", "name": "SO Automatic Sizer",
"description": "A high-speed, high-precision device for sorting round
vegetables like potatoes and onions.", "vegetable_type": ["potato",
"onion"], "function": "Sorting", "category": "Sorting & Grading",
    "models": [
        { "model_id": "SO-90", "input_capacity_kg_h": 15000,
"power_kw": 1.5, "power_supply": "400V", "dimensions_m": { "length":
3.5, "width": 1.2, "height": 2.2 }, "net_weight_kg": 900 }
    ], "image_url":
"https://placehold.co/600x400/faa307/ffffff?text=Auto+Sizer"
  }
];

// Updated line definitions based on catalog
const processingLines = {
    potato: {
```

```
        name: "Complete Potato Washing & Packing Line",
        machines: ["KMK-MD-WASHER", "KMK-S-BRUSH-CLEANER",
"KMK-SO-SIZER", "KMK-WR-SERIES"],
        description: "A full line for washing, dry cleaning, sorting,
and weighing potatoes for market.",
    },
    onion: {
        name: "Complete Onion Topping & Packing Line",
        machines: ["KMK-OS-TOPPER", "KMK-S-BRUSH-CLEANER",
"KMK-SO-SIZER", "KMK-WK-SERIES"],
        description: "A full line for topping, cleaning, sorting, and
weighing onions efficiently.",
    }
};

// Translations with new entries
const translations = {
    en: {
        title: "KMK Machine Selector",
        home: "Home", selector: "Machine Selector", contact:
"Contact",
        tagline: "Find the perfect machinery for your vegetable
processing needs.",
        getStarted: "Get Started",
        warehouseSize: "Warehouse Size (m²)",
        vegetable: "Vegetable", potato: "Potato", onion: "Onion",
        lineOrIndividual: "Processing Type", fullLine: "Full Line",
individualMachines: "Individual Machines",
        findMachines: "Find Machines",
        results: "Results", recommendedLine: "Recommended Line",
compatibleMachines: "Compatible Individual Machines",
        noResults: "No compatible machines found for your criteria.",
        insufficientSpace: "Warning: Your warehouse space may be
insufficient for the recommended line.",
        totalArea: "Total Area Required", totalCapacity: "Estimated
Line Capacity",
        details: "View Details", getOffer: "Get Offer",
        selectModel: "Select Model", productDetails: "Product
Details", backToResults: "Back to Results",
        contactUs: "Contact Us", contactIntro: "Have questions or need
a custom quote? Reach out to our sales team.",
        emailSales: "Email Sales Team",
        footerText: "© 2025 KMK Maszyny. All rights reserved.",
        selectVege: "Select Vegetable", selectType: "Select Type",
        specifications: "Specifications", capacity: "Capacity", power:
"Power", dimensions: "Dimensions", weight: "Weight",
        capacityCalculator: "Capacity Calculator", hoursPerDay: "Hours
per day", calculate: "Calculate", totalOutput: "Estimated Daily
```

```
Output",
    },
    pl: {
        title: "Selektor Maszyn KMK",
        home: "Strona Główna", selector: "Selektor Maszyn", contact:
"Kontakt",
        tagline: "Znajdź idealne maszyny do swoich potrzeb w
przetwórstwie warzyw.",
        getStarted: "Rozpocznij",
        warehouseSize: "Powierzchnia magazynu (m²)",
        vegetable: "Warzywo", potato: "Ziemniak", onion: "Cebula",
        lineOrIndividual: "Typ Przetwarzania", fullLine: "Pełna
Linia", individualMachines: "Pojedyncze Maszyny",
        findMachines: "Znajdź Maszyny",
        results: "Wyniki", recommendedLine: "Rekomendowana Linia",
compatibleMachines: "Kompatybilne Pojedyncze Maszyny",
        noResults: "Nie znaleziono kompatybilnych maszyn dla podanych
kryteriów.",
        insufficientSpace: "Ostrzeżenie: Powierzchnia Twojego magazynu
może być niewystarczająca dla rekomendowanej linii.",
        totalArea: "Całkowita wymagana powierzchnia", totalCapacity:
"Szacowana wydajność linii",
        details: "Zobacz szczegóły", getOffer: "Uzyskaj Ofertę",
        selectModel: "Wybierz Model", productDetails: "Szczegóły
Produktu", backToResults: "Powrót do Wyników",
        contactUs: "Skontaktuj się z nami", contactIntro: "Masz
pytania lub potrzebujesz indywidualnej wyceny? Skontaktuj się z naszym
zespołem sprzedaży.",
        emailSales: "Wyślij e-mail do zespołu sprzedaży",
        footerText: "© 2025 KMK Maszyny. Wszelkie prawa zastrzeżone.",
        selectVege: "Wybierz Warzywo", selectType: "Wybierz Typ",
        specifications: "Specyfikacje", capacity: "Wydajność", power:
"Moc", dimensions: "Wymiary", weight: "Waga",
        capacityCalculator: "Kalkulator wydajności", hoursPerDay:
"Godzin dziennie", calculate: "Oblicz", totalOutput: "Szacowana
dzienna produkcja",
    },
    fr: {
        title: "Sélecteur de Machines KMK",
        home: "Accueil", selector: "Sélecteur de Machines", contact:
"Contact",
        tagline: "Trouvez la machinerie parfaite pour vos besoins de
traitement de légumes.",
        getStarted: "Commencer",
        warehouseSize: "Taille de l'entrepôt (m²)",
        vegetable: "Légume", potato: "Pomme de terre", onion:
"Oignon",
        lineOrIndividual: "Type de Traitement", fullLine: "Ligne
```

```
Complète", individualMachines: "Machines Individuelles",
        findMachines: "Trouver des Machines",
        results: "Résultats", recommendedLine: "Ligne Recommandée",
compatibleMachines: "Machines Individuelles Compatibles",
        noResults: "Aucune machine compatible trouvée pour vos
critères.",
        insufficientSpace: "Avertissement : L'espace de votre entrepôt
peut être insuffisant pour la ligne recommandée.",
        totalArea: "Surface Totale Requise", totalCapacity: "Capacité
de Ligne Estimée",
        details: "Voir les détails", getOffer: "Obtenir une Offre",
        selectModel: "Sélectionner le Modèle", productDetails:
"Détails du Produit", backToResults: "Retour aux Résultats",
        contactUs: "Contactez-nous", contactIntro: "Vous avez des
questions ou besoin d'un devis personnalisé ? Contactez notre équipe
de vente.",
        emailSales: "Envoyer un e-mail à l'équipe de vente",
        footerText: "© 2025 KMK Maszyny. Tous droits réservés.",
        selectVege: "Sélectionnez un légume", selectType:
"Sélectionnez le type",
        specifications: "Spécifications", capacity: "Capacité", power:
"Puissance", dimensions: "Dimensions", weight: "Poids",
        capacityCalculator: "Calculateur de capacité", hoursPerDay:
"Heures par jour", calculate: "Calculer", totalOutput: "Production
journalière estimée",
    }
};

// Main App Component
export default function App() {
    const [page, setPage] = useState('home');
    const [language, setLanguage] = useState('en');
    const [results, setResults] = useState(null);
    const [selectedProduct, setSelectedProduct] = useState(null);
    const [isMobileMenuOpen, setIsMobileMenuOpen] = useState(false);

    const t = translations[language];

    const navigate = (pageName) => {
        setPage(pageName);
        setSelectedProduct(null);
        setResults(null);
        setIsMobileMenuOpen(false);
        window.scrollTo(0, 0);
    };

    const handleSearch = (criteria) => {
        let filteredMachines = [];
```

```
        let recommendedLine = null;
        let totalArea = 0;
        let lineCapacity = 0; // The capacity of a line is determined
by the slowest machine
        let spaceWarning = false;

        if (criteria.lineOrIndividual === 'fullLine' &&
criteria.vegetable) {
            const lineDef = processingLines[criteria.vegetable];
            if (lineDef) {
                recommendedLine = { ...lineDef, machines: [] };
                let capacities = [];
                lineDef.machines.forEach(machineId => {
                    const machine = machineData.find(m => m.machine_id
=== machineId);
                    if (machine) {
                        const model = machine.models[0];
                        recommendedLine.machines.push({ ...machine,
selectedModel: model });
                        totalArea += model.dimensions_m.length *
model.dimensions_m.width;
                        capacities.push(model.input_capacity_kg_h);
                    }
                });
                lineCapacity = Math.min(...capacities); // Line
capacity is bottlenecked by the slowest machine
                recommendedLine.totalArea = totalArea.toFixed(2);
                recommendedLine.lineCapacity = lineCapacity;
                if (criteria.warehouseSize && totalArea >
criteria.warehouseSize) {
                    spaceWarning = true;
                }
            }
        } else {
            filteredMachines = machineData.filter(machine => {
                const vegetableMatch = criteria.vegetable ?
machine.vegetable_type.includes(criteria.vegetable) : true;
                const spaceMatch = criteria.warehouseSize ?
(machine.models.some(m => m.dimensions_m.length * m.dimensions_m.width
<= criteria.warehouseSize)) : true;
                return vegetableMatch && spaceMatch;
            });
        }

        setResults({ filteredMachines, recommendedLine, spaceWarning,
criteria });
        setPage('results');
    };
```

```
    const viewProduct = (product) => {
        setSelectedProduct(product);
        setPage('productDetail');
    };

    const renderPage = () => {
        switch (page) {
            case 'home': return <HomePage t={t} onGetStarted={() =>
navigate('selector')} />;
            case 'selector': return <MachineSelectorPage t={t}
onSearch={handleSearch} />;
            case 'results': return <ResultsPage t={t}
results={results} onViewProduct={viewProduct} />;
            case 'productDetail': return <ProductDetailPage t={t}
product={selectedProduct} onBack={() => setPage('results')} />;
            case 'contact': return <ContactPage t={t} />;
            default: return <HomePage t={t} onGetStarted={() =>
navigate('selector')} />;
        }
    };

    return (
        <div className="bg-gray-100 min-h-screen font-sans
text-gray-800">
            <Header t={t} onNavigate={navigate} language={language}
setLanguage={setLanguage} isMobileMenuOpen={isMobileMenuOpen}
setIsMobileMenuOpen={setIsMobileMenuOpen} />
            <main className="pt-20">
                {renderPage()}
            </main>
            <Footer t={t} />
        </div>
    );
}

// Header Component
function Header({ t, onNavigate, language, setLanguage,
isMobileMenuOpen, setIsMobileMenuOpen }) {
    return (
        <header className="bg-white shadow-lg fixed w-full top-0
z-50">
            <div className="container mx-auto px-4 sm:px-6 lg:px-8">
                <div className="flex justify-between items-center
h-20">
                    <div className="flex-shrink-0 cursor-pointer"
onClick={() => onNavigate('home')}>
                        <h1 className="text-2xl font-bold
```

```
text-orange-600 flex items-center">
                            <Settings className="mr-2" /> {t.title}
                        </h1>
                    </div>
                    <nav className="hidden md:flex items-center
space-x-8 font-medium">
                        <a href="#" className="text-gray-600
hover:text-orange-600 transition" onClick={(e) => {e.preventDefault();
onNavigate('home')}}>{t.home}</a>
                        <a href="#" className="text-gray-600
hover:text-orange-600 transition" onClick={(e) => {e.preventDefault();
onNavigate('selector')}}>{t.selector}</a>
                        <a href="#" className="text-gray-600
hover:text-orange-600 transition" onClick={(e) => {e.preventDefault();
onNavigate('contact')}}>{t.contact}</a>
                    </nav>
                    <div className="flex items-center">
                         <LanguageSwitcher language={language}
setLanguage={setLanguage} t={t} />
                        <div className="md:hidden ml-4">
                            <button onClick={() =>
setIsMobileMenuOpen(!isMobileMenuOpen)} className="text-gray-600
hover:text-orange-600">
                                {isMobileMenuOpen ? <X size={24} /> :
<Menu size={24} />}
                            </button>
                        </div>
                    </div>
                </div>
            </div>
            {isMobileMenuOpen && (
                <div className="md:hidden bg-white border-t">
                    <nav className="flex flex-col items-center
space-y-4 py-4">
                        <a href="#" className="text-gray-600
hover:text-orange-600 transition" onClick={(e) => {e.preventDefault();
onNavigate('home')}}>{t.home}</a>
                        <a href="#" className="text-gray-600
hover:text-orange-600 transition" onClick={(e) => {e.preventDefault();
onNavigate('selector')}}>{t.selector}</a>
                        <a href="#" className="text-gray-600
hover:text-orange-600 transition" onClick={(e) => {e.preventDefault();
onNavigate('contact')}}>{t.contact}</a>
                    </nav>
                </div>
            )}
        </header>
    );
```

```
}

// Language Switcher Component
function LanguageSwitcher({ language, setLanguage }) {
    const [isOpen, setIsOpen] = useState(false);
    const languages = { en: "English", pl: "Polski", fr: "Français" };

    return (
        <div className="relative">
            <button onClick={() => setIsOpen(!isOpen)} className="flex
items-center text-gray-600 hover:text-orange-600 transition">
                <Globe size={20} />
                <span className="ml-2 hidden
sm:inline">{languages[language]}</span>
                <ChevronDown size={16} className={`ml-1
transition-transform ${isOpen ? 'rotate-180' : ''}`} />
            </button>
            {isOpen && (
                <div className="absolute right-0 mt-2 w-32 bg-white
rounded-md shadow-lg z-10 ring-1 ring-black ring-opacity-5">
                    {Object.entries(languages).map(([code, name]) => (
                        <a key={code} href="#" className="block px-4
py-2 text-sm text-gray-700 hover:bg-orange-100"
                           onClick={(e) => { e.preventDefault();
setLanguage(code); setIsOpen(false); }}>
                           {name}
                        </a>
                    ))}
                </div>
            )}
        </div>
    );
}

// Home Page Component
function HomePage({ t, onGetStarted }) {
    return (
        <div className="text-center py-24 md:py-32 px-4 bg-white">
             <div className="max-w-4xl mx-auto">
                <h2 className="text-4xl md:text-6xl font-extrabold
text-gray-900 mb-4">{t.title}</h2>
                <p className="text-lg md:text-xl text-gray-600
max-w-3xl mx-auto mb-10">{t.tagline}</p>
                <button onClick={onGetStarted}
className="bg-orange-600 text-white font-bold py-4 px-10 rounded-full
hover:bg-orange-700 transition-transform transform hover:scale-105
shadow-xl text-lg">
                    {t.getStarted}
```

```
                </button>
            </div>
        </div>
    );
}

// Machine Selector Page Component
function MachineSelectorPage({ t, onSearch }) {
    const [criteria, setCriteria] = useState({ warehouseSize: '',
vegetable: '', lineOrIndividual: '' });
    const handleChange = (e) => setCriteria(prev => ({ ...prev,
[e.target.name]: e.target.value }));
    const handleSubmit = (e) => { e.preventDefault();
onSearch(criteria); };

    return (
        <div className="container mx-auto px-4 py-12">
            <div className="max-w-2xl mx-auto">
                <h2 className="text-3xl font-bold text-center mb-2
text-gray-900">{t.selector}</h2>
                <p className="text-center text-gray-600 mb-8">Fill in
your requirements to find the best solution.</p>
                <form onSubmit={handleSubmit} className="bg-white p-8
rounded-xl shadow-2xl space-y-6">
                    <div>
                        <label htmlFor="warehouseSize"
className="block text-sm font-bold text-gray-700
mb-2">{t.warehouseSize}</label>
                        <input type="number" name="warehouseSize"
id="warehouseSize" value={criteria.warehouseSize}
onChange={handleChange} className="w-full px-4 py-3 border
border-gray-300 rounded-lg focus:ring-orange-500
focus:border-orange-500" placeholder="e.g., 500" />
                    </div>
                    <div>
                        <label htmlFor="vegetable" className="block
text-sm font-bold text-gray-700 mb-2">{t.vegetable}</label>
                        <select name="vegetable" id="vegetable"
value={criteria.vegetable} onChange={handleChange} className="w-full
px-4 py-3 border border-gray-300 rounded-lg focus:ring-orange-500
focus:border-orange-500" required>
                            <option value="">{t.selectVege}</option>
                            <option value="potato">{t.potato}</option>
                            <option value="onion">{t.onion}</option>
                        </select>
                    </div>
                    <div>
                        <label htmlFor="lineOrIndividual"
```

```
className="block text-sm font-bold text-gray-700
mb-2">{t.lineOrIndividual}</label>
                        <select name="lineOrIndividual"
id="lineOrIndividual" value={criteria.lineOrIndividual}
onChange={handleChange} className="w-full px-4 py-3 border
border-gray-300 rounded-lg focus:ring-orange-500
focus:border-orange-500" required>
                            <option value="">{t.selectType}</option>
                            <option
value="fullLine">{t.fullLine}</option>
                            <option
value="individualMachines">{t.individualMachines}</option>
                        </select>
                    </div>
                    <button type="submit" className="w-full
bg-orange-600 text-white font-bold py-3 px-6 rounded-lg
hover:bg-orange-700 transition flex items-center justify-center
text-lg">
                        <Search className="mr-2" /> {t.findMachines}
                    </button>
                </form>
            </div>
        </div>
    );
}

// Capacity Calculator Component
function CapacityCalculator({ lineCapacity, t }) {
    const [hours, setHours] = useState(8);
    const [totalOutput, setTotalOutput] = useState(null);

    useEffect(() => {
        if (hours > 0) {
            setTotalOutput((lineCapacity * hours) / 1000); // Convert
to tonnes
        } else {
            setTotalOutput(null);
        }
    }, [hours, lineCapacity]);

    return (
        <div className="bg-white p-6 rounded-lg shadow-md mt-6
border-t-4 border-orange-500">
            <h4 className="text-xl font-semibold mb-4 flex
items-center"><Calculator className="mr-2
text-orange-600"/>{t.capacityCalculator}</h4>
            <div className="flex flex-col sm:flex-row items-center
gap-4">
```

```
                <div className="flex-1 w-full">
                    <label htmlFor="hours" className="block text-sm
font-medium text-gray-700">{t.hoursPerDay}</label>
                    <input type="number" id="hours" value={hours}
onChange={e => setHours(e.target.value)} className="w-full mt-1 px-3
py-2 border border-gray-300 rounded-md focus:ring-orange-500
focus:border-orange-500"/>
                </div>
                <div className="text-2xl text-gray-500 hidden
sm:block"><ChevronsRight/></div>
                <div className="flex-1 w-full text-center bg-orange-50
p-4 rounded-lg">
                    <h5 className="font-bold
text-gray-700">{t.totalOutput}</h5>
                    <p className="text-3xl font-bold
text-orange-600">{totalOutput ? `${totalOutput.toFixed(2)} tonnes` :
'...'}</p>
                </div>
            </div>
        </div>
    );
}

// Results Page Component
function ResultsPage({ t, results, onViewProduct }) {
    if (!results) return <div className="text-center
py-12">{t.noResults}</div>;
    const { recommendedLine, filteredMachines, spaceWarning } =
results;

    return (
        <div className="container mx-auto px-4 py-12">
            <h2 className="text-4xl font-extrabold mb-8
text-gray-900">{t.results}</h2>
            {spaceWarning && (
                <div className="bg-yellow-100 border-l-4
border-yellow-500 text-yellow-800 p-4 mb-8 rounded-md" role="alert">
                    <p className="font-bold flex
items-center"><AlertTriangle
className="mr-2"/>{t.insufficientSpace}</p>
                </div>
            )}
            {recommendedLine && (
                <div className="mb-12">
                    <h3 className="text-3xl font-bold mb-4
text-orange-700">{recommendedLine.name}</h3>
                    <div className="bg-white p-6 rounded-lg shadow-lg
mb-6">
```

```
                        <p className="text-gray-600
mb-6">{recommendedLine.description}</p>
                        <div className="grid grid-cols-1
md:grid-cols-2 gap-6 text-center">
                            <div className="bg-gray-100 p-4
rounded-lg">
                                <h4 className="font-bold
text-gray-800">{t.totalArea}</h4>
                                <p className="text-3xl font-bold
text-orange-600">{recommendedLine.totalArea} m²</p>
                            </div>
                            <div className="bg-gray-100 p-4
rounded-lg">
                                <h4 className="font-bold
text-gray-800">{t.totalCapacity}</h4>
                                <p className="text-3xl font-bold
text-orange-600">{recommendedLine.lineCapacity.toLocaleString()}
kg/h</p>
                            </div>
                        </div>
                        <CapacityCalculator
lineCapacity={recommendedLine.lineCapacity} t={t} />
                    </div>
                    <div className="grid grid-cols-1 md:grid-cols-2
lg:grid-cols-3 xl:grid-cols-4 gap-6">
                        {recommendedLine.machines.map(machine =>
<MachineCard key={machine.machine_id} machine={machine} t={t}
onViewProduct={onViewProduct} />)}
                    </div>
                </div>
            )}
            {filteredMachines?.length > 0 && (
                <div>
                    <h3 className="text-3xl font-bold
mb-4">{t.compatibleMachines}</h3>
                    <div className="grid grid-cols-1 md:grid-cols-2
lg:grid-cols-3 xl:grid-cols-4 gap-6">
                        {filteredMachines.map(machine => <MachineCard
key={machine.machine_id} machine={machine} t={t}
onViewProduct={onViewProduct} />)}
                    </div>
                </div>
            )}
            {!recommendedLine && !filteredMachines?.length && (
                 <div className="text-center py-16 bg-white rounded-lg
shadow-md">
                    <p className="text-2xl
text-gray-500">{t.noResults}</p>
```

```
                </div>
            )}
        </div>
    );
}

// Machine Card Component
function MachineCard({ machine, t, onViewProduct }) {
    const defaultModel = machine.selectedModel || machine.models[0];
    const area = (defaultModel.dimensions_m.length *
defaultModel.dimensions_m.width).toFixed(2);

    return (
        <div className="bg-white rounded-lg shadow-lg overflow-hidden
transform hover:-translate-y-2 transition-transform duration-300 flex
flex-col group">
            <div className="relative">
                <img src={machine.image_url} alt={machine.name}
className="w-full h-48 object-cover"/>
                <div className="absolute top-2 right-2 bg-orange-600
text-white text-xs font-bold px-2 py-1
rounded-full">{machine.function}</div>
            </div>
            <div className="p-5 flex flex-col flex-grow">
                <h4 className="text-xl font-bold mb-2
text-gray-900">{machine.name}</h4>
                <p className="text-gray-600 text-sm mb-4
flex-grow">{machine.description.substring(0, 90)}...</p>
                <div className="grid grid-cols-2 gap-4 text-sm mb-5
border-t border-b py-3">
                    <div>
                        <p className="font-semibold
text-gray-500">{t.capacity}</p>
                        <p
className="font-bold">{defaultModel.input_capacity_kg_h.toLocaleString
()} kg/h</p>
                    </div>
                    <div>
                        <p className="font-semibold
text-gray-500">Area</p>
                        <p className="font-bold">{area} m²</p>
                    </div>
                </div>
                <button onClick={() => onViewProduct(machine)}
className="mt-auto w-full bg-orange-500 text-white font-bold py-2 px-4
rounded-md hover:bg-orange-600 transition group-hover:bg-orange-600">
                    {t.details}
                </button>
```

```
            </div>
        </div>
    );
}

// Product Detail Page Component
function ProductDetailPage({ t, product, onBack }) {
    const [selectedModelId, setSelectedModelId] =
useState(product.models[0].model_id);
    const selectedModel = product.models.find(m => m.model_id ===
selectedModelId);

    const handleGetOffer = () => {
        const subject = `Offer Request for ${product.name} -
${selectedModel.model_id}`;
        const body = `Hello Sales Team,\n\nI am interested in getting
an offer for the following machine:\n\nMachine:
${product.name}\nModel: ${selectedModel.model_id}\n\nPlease contact me
with more information.\n\nThank you,`;
        window.location.href =
`mailto:akram.khelaifia@kmk-maszyny.com?subject=${encodeURIComponent(s
ubject)}&body=${encodeURIComponent(body)}`;
    };

    return (
        <div className="container mx-auto px-4 py-12">
            <button onClick={onBack} className="mb-8 text-orange-600
hover:text-orange-800 font-bold flex items-center">
                ← {t.backToResults}
            </button>
            <div className="bg-white rounded-xl shadow-2xl
overflow-hidden">
                <div className="grid grid-cols-1 lg:grid-cols-5">
                    <div className="lg:col-span-3">
                        <img src={product.image_url}
alt={product.name} className="w-full h-full object-cover
min-h-[300px]"/>
                    </div>
                    <div className="p-8 lg:col-span-2">
                        <span className="bg-orange-100 text-orange-800
text-sm font-medium mr-2 px-2.5 py-0.5
rounded-full">{product.category}</span>
                        <h2 className="text-4xl font-extrabold my-3
text-gray-900">{product.name}</h2>
                        <p className="text-gray-600
mb-6">{product.description}</p>

                        <div className="mb-6">
```

```
                            <label htmlFor="model" className="block
text-sm font-bold text-gray-700 mb-2">{t.selectModel}</label>
                            <select id="model" value={selectedModelId}
onChange={(e) => setSelectedModelId(e.target.value)} className="w-full
px-4 py-3 border border-gray-300 rounded-lg focus:ring-orange-500
focus:border-orange-500">
                                {product.models.map(model => <option
key={model.model_id}
value={model.model_id}>{model.model_id}</option>)}
                            </select>
                        </div>

                        <div className="bg-gray-50 p-4 rounded-lg mb-6
border">
                            <h4 className="font-bold text-lg
mb-3">{t.specifications}</h4>
                            <div className="space-y-3 text-sm">
                                <div className="flex justify-between
items-center"><span className="flex items-center
text-gray-600"><ChevronsRight className="w-4 h-4 mr-2
text-orange-500"/>{t.capacity}</span> <span
className="font-bold">{selectedModel.input_capacity_kg_h.toLocaleStrin
g()} kg/h</span></div>
                                <div className="flex justify-between
items-center"><span className="flex items-center text-gray-600"><Zap
className="w-4 h-4 mr-2 text-orange-500"/>{t.power}</span> <span
className="font-bold">{selectedModel.power_kw} kW
({selectedModel.power_supply})</span></div>
                                <div className="flex justify-between
items-center"><span className="flex items-center text-gray-600"><Ruler
className="w-4 h-4 mr-2 text-orange-500"/>{t.dimensions}</span> <span
className="font-bold">{selectedModel.dimensions_m.length}x{selectedMod
el.dimensions_m.width}x{selectedModel.dimensions_m.height}
m</span></div>
                                <div className="flex justify-between
items-center"><span className="flex items-center
text-gray-600"><Weight className="w-4 h-4 mr-2
text-orange-500"/>{t.weight}</span> <span
className="font-bold">{selectedModel.net_weight_kg} kg</span></div>
                            </div>
                        </div>

                        <button onClick={handleGetOffer}
className="w-full bg-orange-600 text-white font-bold py-3 px-6
rounded-lg hover:bg-orange-700 transition flex items-center
justify-center text-lg">
                            <Mail className="mr-2" /> {t.getOffer}
                        </button>
```

```
                    </div>
                </div>
            </div>
        </div>
    );
}

// Contact Page Component
function ContactPage({ t }) {
     const handleEmail = () => {
        const subject = `General Inquiry from Website`;
        const body = `Hello Sales Team,\n\nI have a question...\n\n`;
        window.location.href =
`mailto:akram.khelaifia@kmk-maszyny.com?subject=${encodeURIComponent(s
ubject)}&body=${encodeURIComponent(body)}`;
    };
    return (
        <div className="bg-white">
            <div className="container mx-auto px-4 py-24 text-center">
                <h2 className="text-4xl font-extrabold mb-4
text-gray-900">{t.contactUs}</h2>
                <p className="text-lg text-gray-600 mb-8 max-w-2xl
mx-auto">{t.contactIntro}</p>
                <button onClick={handleEmail} className="bg-orange-600
text-white font-bold py-4 px-10 rounded-full hover:bg-orange-700
transition-transform transform hover:scale-105 shadow-xl flex
items-center justify-center mx-auto text-lg">
                    <Mail className="mr-2" /> {t.emailSales}
                </button>
            </div>
        </div>
    );
}

// Footer Component
function Footer({ t }) {
    return (
        <footer className="bg-gray-800 text-gray-400">
            <div className="container mx-auto py-6 px-4 sm:px-6
lg:px-8 text-center">
                <p>{t.footerText}</p>
            </div>
        </footer>
    );
}
```