```html
<!DOCTYPE html>
<html lang="en" class="h-full">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>CropWise - Agricultural Crop Guide</title>
    <script src="https://cdn.tailwindcss.com"></script>
    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
    <script>
        tailwind.config = {
            theme: {
                extend: {
                    colors: {
                        primary: '#5D5CDE',
                        green: {
                            50: '#f0fdf4',
                            100: '#dcfce7',
                            200: '#bbf7d0',
                            300: '#86efac',
                            400: '#4ade80',
                            500: '#22c55e',
                            600: '#16a34a',
                            700: '#15803d',
                            800: '#166534',
                            900: '#14532d'
                        }
                    }
                }
            }
        }
    </script>
    <style>
        .crop-card {
            transition: all 0.3s ease;
        }
        .crop-card:hover {
            transform: translateY(-2px);
        }
        .filter-chip {
            transition: all 0.2s ease;
        }
        .filter-chip:hover {
            transform: scale(1.05);
        }
        .drag-drop-area {
            transition: all 0.3s ease;
        }
        .drag-drop-area.dragover {
            background-color: rgba(93, 92, 222, 0.1);
            border-color: #5D5CDE;
        }
        .pulse-animation {
            animation: pulse 2s cubic-bezier(0.4, 0, 0.6, 1) infinite;
        }
    </style>
</head>
<body class="h-full bg-white dark:bg-gray-900 text-gray-900 dark:text-white">
    <!-- Header -->
    <header class="bg-white dark:bg-gray-800 shadow-sm border-b border-gray-200 dark:border-
gray-700 sticky top-0 z-50">
```

```
        <div class="max-w-7xl mx-auto px-4 sm:px-6 lg:px-8">
            <div class="flex items-center justify-between h-16">
                <div class="flex items-center space-x-3">
                    <div class="w-10 h-10 bg-primary rounded-lg flex items-center justify-center">
                        <span class="text-white font-bold text-lg">🌾</span>
                    </div>
                    <h1 class="text-xl font-bold text-gray-900 dark:text-white">CropWise</h1>
                </div>
                <div class="text-sm text-gray-500 dark:text-gray-400">
                    <span id="cropCount">20</span> crops available
                </div>
            </div>
        </div>
    </header>

    <div class="max-w-7xl mx-auto px-4 sm:px-6 lg:px-8 py-6">
        <!-- Navigation Tabs -->
        <div class="mb-8">
            <div class="border-b border-gray-200 dark:border-gray-700">
                <nav class="-mb-px flex space-x-8">
                    <button id="browseTab" class="tab-button py-2 px-1 border-b-2 border-primary
text-primary font-medium text-sm whitespace-nowrap" onclick="switchTab('browse')">
                        🌾 Browse Crops
                    </button>
                    <button id="identifyTab" class="tab-button py-2 px-1 border-b-2 border-transparent
text-gray-500 dark:text-gray-400 hover:text-gray-700 dark:hover:text-gray-300 hover:border-
gray-300 dark:hover:border-gray-600 font-medium text-sm whitespace-nowrap"
onclick="switchTab('identify')">
                        📷 Identify Crop
                    </button>
                    <button id="analyticsTab" class="tab-button py-2 px-1 border-b-2 border-
transparent text-gray-500 dark:text-gray-400 hover:text-gray-700 dark:hover:text-gray-300
hover:border-gray-300 dark:hover:border-gray-600 font-medium text-sm whitespace-nowrap"
onclick="switchTab('analytics')">
                        📊 Visual Analytics
                    </button>
                    <button id="compareTab" class="tab-button py-2 px-1 border-b-2 border-
transparent text-gray-500 dark:text-gray-400 hover:text-gray-700 dark:hover:text-gray-300
hover:border-gray-300 dark:hover:border-gray-600 font-medium text-sm whitespace-nowrap"
onclick="switchTab('compare')">
                        ⚖️ Compare Crops
                    </button>
                </nav>
            </div>
        </div>

        <!-- Browse Section -->
        <div id="browseSection">
            <!-- Search and Filters -->
            <div class="mb-8">
                <!-- Search Bar -->
                <div class="relative mb-6">
                    <input
                        type="text"
                        id="searchInput"
                        placeholder="Search crops by name..."
```

```
                    class="w-full px-4 py-3 text-base border border-gray-300 dark:border-gray-600
rounded-lg focus:ring-2 focus:ring-primary focus:border-transparent dark:bg-gray-800 dark:text-
white"
                />
                <div class="absolute inset-y-0 right-0 flex items-center pr-3">
                    <svg class="h-5 w-5 text-gray-400" fill="none" stroke="currentColor" viewBox="0
0 24 24">
                        <path stroke-linecap="round" stroke-linejoin="round" stroke-width="2" d="M21
21l-6-6m2-5a7 7 0 11-14 0 7 7 0 0114 0z"></path>
                    </svg>
                </div>
            </div>

            <!-- Filter Chips -->
            <div class="flex flex-wrap gap-2 mb-4">
                <button class="filter-chip px-4 py-2 bg-gray-100 dark:bg-gray-700 text-gray-700
dark:text-gray-300 rounded-full text-sm font-medium hover:bg-primary hover:text-white
transition-colors" data-filter="all">
                    All Crops
                </button>
                <button class="filter-chip px-4 py-2 bg-gray-100 dark:bg-gray-700 text-gray-700
dark:text-gray-300 rounded-full text-sm font-medium hover:bg-primary hover:text-white
transition-colors" data-filter="low-water">
                    Low Water Needs
                </button>
                <button class="filter-chip px-4 py-2 bg-gray-100 dark:bg-gray-700 text-gray-700
dark:text-gray-300 rounded-full text-sm font-medium hover:bg-primary hover:text-white
transition-colors" data-filter="drought-resistant">
                    Drought Resistant
                </button>
                <button class="filter-chip px-4 py-2 bg-gray-100 dark:bg-gray-700 text-gray-700
dark:text-gray-300 rounded-full text-sm font-medium hover:bg-primary hover:text-white
transition-colors" data-filter="intercropping">
                    Intercropping
                </button>
                <button class="filter-chip px-4 py-2 bg-gray-100 dark:bg-gray-700 text-gray-700
dark:text-gray-300 rounded-full text-sm font-medium hover:bg-primary hover:text-white
transition-colors" data-filter="rotation">
                    Crop Rotation
                </button>
            </div>
        </div>

        <!-- Crops Grid -->
        <div id="cropsGrid" class="grid grid-cols-1 md:grid-cols-2 lg:grid-cols-3 gap-6">
            <!-- Crops will be dynamically inserted here -->
        </div>

        <!-- No Results Message -->
        <div id="noResults" class="hidden text-center py-12">
            <div class="text-gray-400 text-6xl mb-4">🔍</div>
            <h3 class="text-lg font-medium text-gray-500 dark:text-gray-400 mb-2">No crops
found</h3>
            <p class="text-gray-400 dark:text-gray-500">Try adjusting your search terms or filters</
p>
        </div>
    </div>

    <!-- Crop Identification Section -->
    <div id="identifySection" class="hidden">
```

```
<div class="max-w-4xl mx-auto">
    <div class="text-center mb-8">
        <h2 class="text-2xl font-bold text-gray-900 dark:text-white mb-4">🔍 Identify Your Crop</h2>
        <p class="text-gray-600 dark:text-gray-400">Upload a photo of your crop and our AI will help identify it from our database</p>
    </div>

    <!-- Upload Area -->
    <div id="uploadArea" class="drag-drop-area border-2 border-dashed border-gray-300 dark:border-gray-600 rounded-lg p-8 text-center mb-6 bg-gray-50 dark:bg-gray-800">
        <div class="space-y-4">
            <div class="text-4xl">📷</div>
            <div>
                <h3 class="text-lg font-medium text-gray-900 dark:text-white mb-2">Upload Crop Image</h3>
                <p class="text-gray-600 dark:text-gray-400 mb-4">Drag and drop an image here, or click to select</p>
                <input type="file" id="imageInput" accept="image/*" class="hidden">
                <button onclick="document.getElementById('imageInput').click()" class="bg-primary text-white px-6 py-3 rounded-lg font-medium hover:bg-purple-700 transition-colors">
                    Choose Image
                </button>
            </div>
            <p class="text-sm text-gray-500 dark:text-gray-400">Supports JPG, PNG, WebP formats</p>
        </div>
    </div>

    <!-- Image Preview -->
    <div id="imagePreview" class="hidden mb-6">
        <div class="bg-white dark:bg-gray-800 rounded-lg shadow-md p-6 border border-gray-200 dark:border-gray-700">
            <h3 class="text-lg font-medium text-gray-900 dark:text-white mb-4">Uploaded Image</h3>
            <div class="flex items-start space-x-6">
                <img id="previewImage" class="w-64 h-48 object-cover rounded-lg border border-gray-200 dark:border-gray-600" />
                <div class="flex-1">
                    <button id="identifyButton" onclick="identifyCrop()" class="bg-primary text-white px-6 py-3 rounded-lg font-medium hover:bg-purple-700 transition-colors mb-4">
                        🔍 Identify Crop
                    </button>
                    <p class="text-sm text-gray-600 dark:text-gray-400">Click "Identify Crop" to analyze your image using AI vision technology.</p>
                </div>
            </div>
        </div>
    </div>

    <!-- Identification Results -->
    <div id="identificationResults" class="hidden">
        <div class="bg-white dark:bg-gray-800 rounded-lg shadow-md p-6 border border-gray-200 dark:border-gray-700">
            <h3 class="text-lg font-medium text-gray-900 dark:text-white mb-4">🎯 Identification Results</h3>
            <div id="resultsContent">
                <!-- Results will be populated here -->
```

```
                </div>
            </div>
        </div>

        <!-- Loading State -->
        <div id="identificationLoading" class="hidden">
            <div class="bg-white dark:bg-gray-800 rounded-lg shadow-md p-8 border border-gray-200 dark:border-gray-700 text-center">
                <div class="pulse-animation text-4xl mb-4">🔍</div>
                <h3 class="text-lg font-medium text-gray-900 dark:text-white mb-2">Analyzing Your Crop...</h3>
                <p class="text-gray-600 dark:text-gray-400">Our AI is examining the image to identify the crop type</p>
                <div class="mt-6">
                    <div class="bg-gray-200 dark:bg-gray-700 rounded-full h-2 overflow-hidden">
                        <div class="bg-primary h-full rounded-full animate-pulse"></div>
                    </div>
                </div>
            </div>
        </div>
    </div>
</div>

<!-- Analytics Section -->
<div id="analyticsSection" class="hidden">
    <div class="space-y-8">
        <div class="text-center">
            <h2 class="text-2xl font-bold text-gray-900 dark:text-white mb-4">📊 Crop Analytics Dashboard</h2>
            <p class="text-gray-600 dark:text-gray-400">Visual insights into crop characteristics and environmental needs</p>
        </div>

        <div class="grid grid-cols-1 lg:grid-cols-2 gap-8">
            <!-- Rainfall Distribution Chart -->
            <div class="bg-white dark:bg-gray-800 rounded-lg shadow-md p-6 border border-gray-200 dark:border-gray-700">
                <h3 class="text-lg font-semibold text-gray-900 dark:text-white mb-4">💧 Rainfall Requirements</h3>
                <canvas id="rainfallChart" width="400" height="300"></canvas>
            </div>

            <!-- Temperature Distribution Chart -->
            <div class="bg-white dark:bg-gray-800 rounded-lg shadow-md p-6 border border-gray-200 dark:border-gray-700">
                <h3 class="text-lg font-semibold text-gray-900 dark:text-white mb-4">🌡️ Temperature Ranges</h3>
                <canvas id="temperatureChart" width="400" height="300"></canvas>
            </div>

            <!-- Cropping Techniques Chart -->
            <div class="bg-white dark:bg-gray-800 rounded-lg shadow-md p-6 border border-gray-200 dark:border-gray-700">
                <h3 class="text-lg font-semibold text-gray-900 dark:text-white mb-4">🌱 Cropping Techniques</h3>
                <canvas id="techniquesChart" width="400" height="300"></canvas>
            </div>
```

```
                <!-- Soil Types Chart -->
                <div class="bg-white dark:bg-gray-800 rounded-lg shadow-md p-6 border border-
gray-200 dark:border-gray-700">
                    <h3 class="text-lg font-semibold text-gray-900 dark:text-white mb-4">⛰️ Soil
Preferences</h3>
                    <canvas id="soilChart" width="400" height="300"></canvas>
                </div>
            </div>
        </div>
    </div>

    <!-- Compare Section -->
    <div id="compareSection" class="hidden">
        <div class="text-center mb-8">
            <h2 class="text-2xl font-bold text-gray-900 dark:text-white mb-4">⚖️ Compare
Crops</h2>
            <p class="text-gray-600 dark:text-gray-400">Select up to 3 crops to compare their
characteristics side by side</p>
        </div>

        <!-- Crop Selection -->
        <div class="mb-8">
            <h3 class="text-lg font-medium text-gray-900 dark:text-white mb-4">Select Crops to
Compare:</h3>
            <div class="grid grid-cols-2 md:grid-cols-3 lg:grid-cols-4 gap-3">
                <div id="compareSelection">
                    <!-- Crop selection buttons will be populated here -->
                </div>
            </div>
        </div>

        <!-- Comparison Results -->
        <div id="comparisonResults" class="hidden">
            <div class="bg-white dark:bg-gray-800 rounded-lg shadow-md p-6 border border-
gray-200 dark:border-gray-700">
                <h3 class="text-lg font-semibold text-gray-900 dark:text-white mb-6">Comparison
Results</h3>
                <div id="comparisonTable">
                    <!-- Comparison table will be populated here -->
                </div>
            </div>
        </div>
    </div>
  </div>

  <!-- Crop Detail Modal -->
  <div id="cropModal" class="hidden fixed inset-0 z-50 overflow-y-auto">
    <div class="flex items-center justify-center min-h-screen px-4 pt-4 pb-20 text-center
sm:block sm:p-0">
        <div class="fixed inset-0 transition-opacity" onclick="closeCropModal()">
            <div class="absolute inset-0 bg-gray-500 dark:bg-gray-900 opacity-75"></div>
        </div>
        <div class="inline-block align-bottom bg-white dark:bg-gray-800 rounded-lg px-4 pt-5
pb-4 text-left overflow-hidden shadow-xl transform transition-all sm:my-8 sm:align-middle
sm:max-w-4xl sm:w-full sm:p-6">
            <div class="flex justify-between items-start mb-4">
                <h2 id="modalTitle" class="text-2xl font-bold text-gray-900 dark:text-white"></h2>
                <button onclick="closeCropModal()" class="text-gray-400 hover:text-gray-600
dark:hover:text-gray-300">
```

```
                <svg class="w-6 h-6" fill="none" stroke="currentColor" viewBox="0 0 24 24">
                  <path stroke-linecap="round" stroke-linejoin="round" stroke-width="2" d="M6
18L18 6M6 6l12 12"></path>
                </svg>
              </button>
            </div>
            <div id="modalContent" class="space-y-6">
              <!-- Modal content will be dynamically inserted here -->
            </div>
          </div>
        </div>
      </div>

      <script>
        // Crop data
        const cropsData = {
          "crops": [
            {
              "name": "Wheat",
              "environmental_needs": {
                "temp": "10--25°C",
                "rainfall": "30--90 cm",
                "sunlight": "Moderate to high",
                "soil": "Well-drained loamy or clayey",
                "humidity": "Low to moderate",
                "weather_factors": ["Rainfall", "Sunlight", "Temperature", "Humidity"]
              },
              "cropping_techniques": [
                "Crop Rotation: With legumes (gram, lentil)",
                "Mixed Cropping: With mustard, gram",
                "Sequential Cropping: After paddy",
                "Relay Cropping: Wheat sown in standing rice",
                "Intercropping: Wheat + chickpea",
                "Strip Cropping: On slopes to reduce erosion"
              ]
            },
            {
              "name": "Rice",
              "environmental_needs": {
                "temp": "20--35°C",
                "rainfall": "100--200 cm",
                "sunlight": "High",
                "soil": "Clayey, water-retentive",
                "humidity": "High",
                "weather_factors": ["Heavy rainfall", "Humidity", "High temperature"]
              },
              "cropping_techniques": [
                "Monocropping: In water-rich regions",
                "Sequential Cropping: Rice → wheat → mung",
                "Relay Cropping: Wheat sown before rice harvest",
                "Mixed Cropping: Rice + black gram",
                "Intercropping: Rice + groundnut (upland areas)"
              ]
            },
            {
              "name": "Maize (Corn)",
              "environmental_needs": {
                "temp": "18--27°C",
                "rainfall": "50--100 cm",
                "sunlight": "Full",
```

"soil": "Loamy, well-drained",
"humidity": "Moderate",
"weather_factors": ["Sunlight", "Temperature", "Moderate rainfall"]
},
"cropping_techniques": [
"Intercropping: Maize + beans/cowpea",
"Crop Rotation: With legumes or mustard",
"Mixed Cropping: Maize + soybean",
"Strip Cropping: Maize + grasses",
"Sequential Cropping: Maize → wheat"
]
},
{
"name": "Sugarcane",
"environmental_needs": {
"temp": "21--27°C",
"rainfall": "75--150 cm",
"sunlight": "Full",
"soil": "Deep loamy",
"humidity": "High",
"weather_factors": ["Humid", "Warm temperature", "Rain"]
},
"cropping_techniques": [
"Intercropping: Sugarcane + onion/garlic/pea",
"Crop Rotation: With pulses/oilseeds",
"Mixed Cropping: Sugarcane + soybean",
"Sequential Cropping: Sugarcane → wheat",
"Agroforestry: Cane + fruit trees"
]
},
{
"name": "Cotton",
"environmental_needs": {
"temp": "21--30°C",
"rainfall": "50--100 cm",
"sunlight": "High",
"soil": "Black soil (regur)",
"humidity": "Low to moderate",
"weather_factors": ["Dry weather", "Sunlight", "Moderate rain"]
},
"cropping_techniques": [
"Intercropping: Cotton + soybean/groundnut",
"Crop Rotation: Cotton → wheat or jowar",
"Mixed Cropping: Cotton + pigeon pea",
"Sequential Cropping: After pulses",
"Strip Cropping: On slopes"
]
},
{
"name": "Barley",
"environmental_needs": {
"temp": "12--25°C",
"rainfall": "30--70 cm",
"sunlight": "Moderate",
"soil": "Sandy loam",
"humidity": "Low",
"weather_factors": ["Cold temperature", "Low humidity"]
},
"cropping_techniques": [

"Crop Rotation: With lentils or chickpeas",
"Mixed Cropping: Barley + mustard",
"Sequential Cropping: After maize",
"Intercropping: Barley + peas"
]
},
{
"name": "Groundnut (Peanut)",
"environmental_needs": {
"temp": "25--30°C",
"rainfall": "50--100 cm",
"sunlight": "Full",
"soil": "Sandy loam",
"humidity": "Moderate",
"weather_factors": ["Warm temperature", "Moderate rainfall"]
},
"cropping_techniques": [
"Intercropping: Groundnut + sunflower",
"Crop Rotation: With cereals (sorghum, maize)",
"Mixed Cropping: Groundnut + castor",
"Sequential Cropping: Groundnut → rabi sorghum"
]
},
{
"name": "Soybean",
"environmental_needs": {
"temp": "20--30°C",
"rainfall": "60--125 cm",
"sunlight": "Full",
"soil": "Loamy or clay",
"humidity": "Moderate",
"weather_factors": ["Rainfall", "Sunlight", "Moderate temperature"]
},
"cropping_techniques": [
"Intercropping: Soybean + maize/sorghum",
"Crop Rotation: Soybean → wheat or rice",
"Mixed Cropping: Soybean + pigeon pea",
"Sequential Cropping: Soybean → chickpea"
]
},
{
"name": "Millet (e.g., Bajra)",
"environmental_needs": {
"temp": "20--30°C",
"rainfall": "40--75 cm",
"sunlight": "Full",
"soil": "Sandy, dry",
"humidity": "Low",
"weather_factors": ["Drought resistance", "Low rainfall"]
},
"cropping_techniques": [
"Intercropping: Millet + cowpea",
"Crop Rotation: With pulses or oilseeds",
"Mixed Cropping: Millet + green gram",
"Sequential Cropping: Millet → sesame"
]
},
{

```
    "name": "Mustard",
    "environmental_needs": {
        "temp": "10--25°C",
        "rainfall": "25--40 cm",
        "sunlight": "Moderate to high",
        "soil": "Loamy",
        "humidity": "Low",
        "weather_factors": ["Cool temperature", "Low humidity"]
    },
    "cropping_techniques": [
        "Mixed Cropping: Mustard + wheat or gram",
        "Crop Rotation: Mustard after rice",
        "Intercropping: Mustard + lentils",
        "Sequential Cropping: Rice → mustard → moong"
    ]
},
{
    "name": "Bambara Groundnut",
    "environmental_needs": {
        "temp": "19--30°C",
        "rainfall": "75--140 cm",
        "soil": "Sandy, well-drained",
        "humidity": "Tolerant",
        "weather_factors": ["Moderate rainfall", "Warm temperature"]
    },
    "cropping_techniques": [
        "Intercropping: With sorghum, millet, maize, peanut, cassava",
        "Monocropping: Also suitable"
    ]
},
{
    "name": "Pearl Millet",
    "environmental_needs": {
        "temp": "25--35°C",
        "rainfall": "30--110 cm",
        "soil": "Sandy, drought-prone",
        "humidity": "Low tolerance",
        "weather_factors": ["Drought", "Low rainfall"]
    },
    "cropping_techniques": [
        "Rotation: With sorghum, groundnut, mustard",
        "Intercropping: With cowpea, mung bean, chickpea"
    ]
},
{
    "name": "Sorghum",
    "environmental_needs": {
        "temp": "25--32°C",
        "rainfall": "40--100 cm",
        "soil": "Sandy loam",
        "humidity": "Moderate",
        "weather_factors": ["Sunlight", "Moderate rainfall"]
    },
    "cropping_techniques": [
        "Rotation: With pearl millet, groundnut, pulses",
        "Intercropping: Sorghum + cowpea/grain pea"
    ]
},
{
    "name": "Potato",
```

```
        "environmental_needs": {
            "temp": "15--20°C",
            "rainfall": "60--120 cm",
            "soil": "Loamy, well-drained",
            "humidity": "Moderate",
            "weather_factors": ["Cool temperature", "Soil moisture"]
        },
        "cropping_techniques": [
            "Rotation: With cereals, legumes to reduce pests"
        ]
    },
    {
        "name": "Cassava",
        "environmental_needs": {
            "temp": "25--35°C",
            "rainfall": "100--150 cm",
            "soil": "Poor, well-drained",
            "humidity": "Tolerant",
            "weather_factors": ["Warm temperature", "Moderate rainfall", "Drought tolerant"]
        },
        "cropping_techniques": [
            "Intercropping: With maize, legumes",
            "Monocropping: In larger scale plantations"
        ]
    },
    {
        "name": "Sunflower",
        "environmental_needs": {
            "temp": "20--26°C",
            "rainfall": "50--75 cm",
            "soil": "Well-drained sandy loam to clay loam",
            "humidity": "Moderate",
            "weather_factors": ["Rainfall timing", "Sunlight", "Temperature"]
        },
        "cropping_techniques": [
            "Crop Rotation: With cereals (maize, sorghum)",
            "Intercropping: With legumes (cowpea, chickpea)"
        ]
    },
    {
        "name": "Sweet Potato",
        "environmental_needs": {
            "temp": "21--30°C",
            "rainfall": "75--100 cm",
            "soil": "Light, well-drained loam",
            "humidity": "Moderate",
            "weather_factors": ["Warm temperature", "Soil moisture"]
        },
        "cropping_techniques": [
            "Crop Rotation: With cereals, legumes"
        ]
    },
    {
        "name": "Carrot",
        "environmental_needs": {
            "temp": "15--21°C",
            "rainfall": "60--80 cm",
            "soil": "Deep sandy loam, free of stones",
            "humidity": "Moderate",
            "weather_factors": ["Cool temperatures", "Even moisture"]
```

```
                },
                "cropping_techniques": [
                    "Crop Rotation: Avoid repeated planting, use cereals/legumes rotation"
                ]
            },
            {
                "name": "Tomato",
                "environmental_needs": {
                    "temp": "18--27°C",
                    "rainfall": "70--120 cm",
                    "soil": "Well-drained loam",
                    "humidity": "Moderate",
                    "weather_factors": ["Warm temperature", "Sunlight", "Moisture for fruiting"]
                },
                "cropping_techniques": [
                    "Crop Rotation: With cereals and legumes"
                ]
            },
            {
                "name": "Banana",
                "environmental_needs": {
                    "temp": "26--30°C",
                    "rainfall": "120--250 cm",
                    "soil": "Rich loam, deep moisture-retentive",
                    "humidity": "High",
                    "weather_factors": ["Tropical humidity", "Rainfall"]
                },
                "cropping_techniques": [
                    "Intercropping: Bananas + legumes, pineapples"
                ]
            }
        ]
    };

    let currentFilter = 'all';
    let searchTerm = '';
    let currentTab = 'browse';
    let selectedCropsForComparison = [];
    let currentUploadedImage = null;

    // Dark mode support
    if (window.matchMedia && window.matchMedia('(prefers-color-scheme: dark)').matches) {
        document.documentElement.classList.add('dark');
    }
    window.matchMedia('(prefers-color-scheme: dark)').addEventListener('change', event => {
        if (event.matches) {
            document.documentElement.classList.add('dark');
        } else {
            document.documentElement.classList.remove('dark');
        }
    });

    // Tab switching functionality
    function switchTab(tabName) {
        // Hide all sections
        document.querySelectorAll('#browseSection, #identifySection, #analyticsSection,
#compareSection').forEach(section => {
            section.classList.add('hidden');
        });
```

```
        // Show selected section
        document.getElementById(tabName + 'Section').classList.remove('hidden');

        // Update tab styles
        document.querySelectorAll('.tab-button').forEach(tab => {
            tab.classList.remove('border-primary', 'text-primary');
            tab.classList.add('border-transparent', 'text-gray-500', 'dark:text-gray-400');
        });

        document.getElementById(tabName + 'Tab').classList.remove('border-transparent', 'text-
gray-500', 'dark:text-gray-400');
        document.getElementById(tabName + 'Tab').classList.add('border-primary', 'text-
primary');

        currentTab = tabName;

        // Initialize tab-specific content
        if (tabName === 'analytics') {
            initializeCharts();
        } else if (tabName === 'compare') {
            initializeCompareSection();
        }
    }

    // Register Poe handler for crop identification
    if (window.Poe && window.Poe.registerHandler) {
        window.Poe.registerHandler("crop-identification", (result, context) => {
            const loadingDiv = document.getElementById('identificationLoading');
            const resultsDiv = document.getElementById('identificationResults');
            const resultsContent = document.getElementById('resultsContent');

            if (result.status === "error") {
                loadingDiv.classList.add('hidden');
                resultsContent.innerHTML = `
                    <div class="text-center text-red-600 dark:text-red-400">
                        <div class="text-4xl mb-4">❌</div>
                        <h4 class="text-lg font-medium mb-2">Identification Failed</h4>
                        <p>Error: ${result.responses[0]?.statusText || 'Unknown error occurred'}</p>
                    </div>
                `;
                resultsDiv.classList.remove('hidden');
            } else if (result.status === "complete") {
                loadingDiv.classList.add('hidden');

                const aiResponse = result.responses[0]?.content || '';
                const identifiedCrops = findMatchingCrops(aiResponse);

                if (identifiedCrops.length > 0) {
                    resultsContent.innerHTML = `
                        <div class="space-y-6">
                            <div class="text-center">
                                <div class="text-4xl mb-4">✅</div>
                                <h4 class="text-lg font-medium text-gray-900 dark:text-white mb-2">Crop
Identified Successfully!</h4>
                            </div>

                            <div class="bg-gray-50 dark:bg-gray-700 rounded-lg p-4 mb-6">
                                <h5 class="font-medium text-gray-900 dark:text-white mb-2">AI
Analysis:</h5>
```

```
                        <p class="text-sm text-gray-700 dark:text-gray-300">${aiResponse}</p>
                    </div>

                    <div class="space-y-4">
                        <h5 class="font-medium text-gray-900 dark:text-white">Possible Matches
from Our Database:</h5>
                        <div class="grid gap-4">
                            ${identifiedCrops.map(crop => `
                                <div class="border border-gray-200 dark:border-gray-600 rounded-lg
p-4 hover:bg-gray-50 dark:hover:bg-gray-700 cursor-pointer transition-colors"
onclick="showCropDetails('${crop.name}')">
                                    <div class="flex items-center space-x-3">
                                        <span class="text-2xl">${getCropEmoji(crop.name)}</span>
                                        <div>
                                            <h6 class="font-medium text-gray-900 dark:text-white">$
{crop.name}</h6>
                                            <p class="text-sm text-gray-600 dark:text-gray-400">
                                                Temp: ${crop.environmental_needs.temp} |
                                                Rainfall: ${crop.environmental_needs.rainfall}
                                            </p>
                                        </div>
                                    </div>
                                </div>
                            `).join('')}
                        </div>
                    </div>
                </div>
            `;
        } else {
            resultsContent.innerHTML = `
                <div class="text-center">
                    <div class="text-4xl mb-4">🤔</div>
                    <h4 class="text-lg font-medium text-gray-900 dark:text-white mb-2">No
Direct Match Found</h4>
                    <div class="bg-gray-50 dark:bg-gray-700 rounded-lg p-4 mb-4">
                        <h5 class="font-medium text-gray-900 dark:text-white mb-2">AI
Analysis:</h5>
                        <p class="text-sm text-gray-700 dark:text-gray-300">${aiResponse}</p>
                    </div>
                    <p class="text-gray-600 dark:text-gray-400 mb-4">The identified crop might
not be in our current database of 20 crops, or the image might need better lighting/angle.</p>
                    <button onclick="resetIdentification()" class="bg-primary text-white px-4 py-2
rounded-lg font-medium hover:bg-purple-700 transition-colors">
                        Try Another Image
                    </button>
                </div>
            `;
        }

        resultsDiv.classList.remove('hidden');
    }
});
}

// Crop identification functionality
function setupImageUpload() {
    const uploadArea = document.getElementById('uploadArea');
    const imageInput = document.getElementById('imageInput');
    const imagePreview = document.getElementById('imagePreview');
    const previewImage = document.getElementById('previewImage');
```

```
    // Drag and drop handlers
    uploadArea.addEventListener('dragover', (e) => {
        e.preventDefault();
        uploadArea.classList.add('dragover');
    });

    uploadArea.addEventListener('dragleave', (e) => {
        e.preventDefault();
        uploadArea.classList.remove('dragover');
    });

    uploadArea.addEventListener('drop', (e) => {
        e.preventDefault();
        uploadArea.classList.remove('dragover');
        const files = e.dataTransfer.files;
        if (files.length > 0) {
            handleImageUpload(files[0]);
        }
    });

    // File input handler
    imageInput.addEventListener('change', (e) => {
        if (e.target.files.length > 0) {
            handleImageUpload(e.target.files[0]);
        }
    });
}

function handleImageUpload(file) {
    if (!file.type.startsWith('image/')) {
        alert('Please upload an image file.');
        return;
    }

    currentUploadedImage = file;
    const reader = new FileReader();
    reader.onload = (e) => {
        document.getElementById('previewImage').src = e.target.result;
        document.getElementById('imagePreview').classList.remove('hidden');
        document.getElementById('identificationResults').classList.add('hidden');
        document.getElementById('identificationLoading').classList.add('hidden');
    };
    reader.readAsDataURL(file);
}

async function identifyCrop() {
    if (!currentUploadedImage) {
        alert('Please upload an image first.');
        return;
    }

    // Show loading state
    document.getElementById('identificationLoading').classList.remove('hidden');
    document.getElementById('identificationResults').classList.add('hidden');

    try {
        if (window.Poe && window.Poe.sendUserMessage) {
            await window.Poe.sendUserMessage(
```

```
                "@Claude-Sonnet-4 Please analyze this crop image and identify the type of crop. Be specific about the crop name and mention any distinctive features you observe. Compare it with common agricultural crops like wheat, rice, maize, cotton, etc.",
                {
                    attachments: [currentUploadedImage],
                    handler: "crop-identification",
                    stream: false,
                    openChat: false
                }
            );
        } else {
            // Fallback for demo purposes
            setTimeout(() => {
                document.getElementById('identificationLoading').classList.add('hidden');
                document.getElementById('resultsContent').innerHTML = `
                    <div class="text-center text-yellow-600 dark:text-yellow-400">
                        <div class="text-4xl mb-4">⚠️</div>
                        <h4 class="text-lg font-medium mb-2">Demo Mode</h4>
                        <p>AI crop identification requires the Poe platform. This is a demo version.</p>
                    </div>
                `;
                document.getElementById('identificationResults').classList.remove('hidden');
            }, 2000);
        }
    } catch (error) {
        document.getElementById('identificationLoading').classList.add('hidden');
        document.getElementById('resultsContent').innerHTML = `
            <div class="text-center text-red-600 dark:text-red-400">
                <div class="text-4xl mb-4">❌</div>
                <h4 class="text-lg font-medium mb-2">Identification Failed</h4>
                <p>Error: ${error.message}</p>
            </div>
        `;
        document.getElementById('identificationResults').classList.remove('hidden');
    }
}

function findMatchingCrops(aiResponse) {
    const response = aiResponse.toLowerCase();
    const matches = [];

    cropsData.crops.forEach(crop => {
        const cropName = crop.name.toLowerCase();
        const cropWords = cropName.split(/[\s(),]+/);

        // Check if crop name or major words appear in the AI response
        if (cropWords.some(word => word.length > 2 && response.includes(word))) {
            matches.push(crop);
        }
    });

    return matches.slice(0, 3); // Return top 3 matches
}

function resetIdentification() {
    currentUploadedImage = null;
    document.getElementById('imagePreview').classList.add('hidden');
    document.getElementById('identificationResults').classList.add('hidden');
```

```
        document.getElementById('identificationLoading').classList.add('hidden');
        document.getElementById('imageInput').value = '';
    }

    // Analytics functionality
    function initializeCharts() {
        // Only initialize if charts haven't been created yet
        if (document.getElementById('rainfallChart').getContext('2d').chart) {
            return;
        }

        createRainfallChart();
        createTemperatureChart();
        createTechniquesChart();
        createSoilChart();
    }

    function createRainfallChart() {
        const ctx = document.getElementById('rainfallChart').getContext('2d');

        // Process rainfall data
        const rainfallCategories = { 'Low (0-50cm)': 0, 'Medium (50-100cm)': 0, 'High (100cm+)':
0 };

        cropsData.crops.forEach(crop => {
            const match = crop.environmental_needs.rainfall.match(/(\d+)--(\d+)/);
            if (match) {
                const avg = (parseInt(match[1]) + parseInt(match[2])) / 2;
                if (avg <= 50) rainfallCategories['Low (0-50cm)']++;
                else if (avg <= 100) rainfallCategories['Medium (50-100cm)']++;
                else rainfallCategories['High (100cm+)']++;
            }
        });

        ctx.chart = new Chart(ctx, {
            type: 'doughnut',
            data: {
                labels: Object.keys(rainfallCategories),
                datasets: [{
                    data: Object.values(rainfallCategories),
                    backgroundColor: ['#FEF3C7', '#DBEAFE', '#BFDBFE'],
                    borderColor: ['#F59E0B', '#3B82F6', '#1D4ED8'],
                    borderWidth: 2
                }]
            },
            options: {
                responsive: true,
                maintainAspectRatio: false,
                plugins: {
                    legend: {
                        position: 'bottom',
                        labels: {
                            color: document.documentElement.classList.contains('dark') ? '#E5E7EB' :
'#374151'
                        }
                    }
                }
            }
        });
    }
```

```
function createTemperatureChart() {
    const ctx = document.getElementById('temperatureChart').getContext('2d');

    // Process temperature data
    const tempRanges = [];
    const cropNames = [];

    cropsData.crops.forEach(crop => {
        const match = crop.environmental_needs.temp.match(/(\d+)--(\d+)/);
        if (match) {
            const min = parseInt(match[1]);
            const max = parseInt(match[2]);
            tempRanges.push({ min, max, name: crop.name });
            cropNames.push(crop.name.split(' ')[0]); // First word only for readability
        }
    });

    ctx.chart = new Chart(ctx, {
        type: 'bar',
        data: {
            labels: cropNames.slice(0, 10), // Show first 10 crops
            datasets: [{
                label: 'Min Temp (°C)',
                data: tempRanges.slice(0, 10).map(r => r.min),
                backgroundColor: '#93C5FD'
            }, {
                label: 'Max Temp (°C)',
                data: tempRanges.slice(0, 10).map(r => r.max),
                backgroundColor: '#F87171'
            }]
        },
        options: {
            responsive: true,
            maintainAspectRatio: false,
            scales: {
                y: {
                    beginAtZero: true,
                    ticks: {
                        color: document.documentElement.classList.contains('dark') ? '#E5E7EB' :
'#374151'
                    }
                },
                x: {
                    ticks: {
                        color: document.documentElement.classList.contains('dark') ? '#E5E7EB' :
'#374151'
                    }
                }
            },
            plugins: {
                legend: {
                    labels: {
                        color: document.documentElement.classList.contains('dark') ? '#E5E7EB' :
'#374151'
                    }
                }
            }
        }
    });
```

```
    }

    function createTechniquesChart() {
        const ctx = document.getElementById('techniquesChart').getContext('2d');

        // Count cropping techniques
        const techniques = {};
        cropsData.crops.forEach(crop => {
            crop.cropping_techniques.forEach(technique => {
                const type = technique.split(':')[0];
                techniques[type] = (techniques[type] || 0) + 1;
            });
        });

        ctx.chart = new Chart(ctx, {
            type: 'bar',
            data: {
                labels: Object.keys(techniques),
                datasets: [{
                    label: 'Number of Crops',
                    data: Object.values(techniques),
                    backgroundColor: '#5D5CDE',
                    borderColor: '#4C46C7',
                    borderWidth: 1
                }]
            },
            options: {
                responsive: true,
                maintainAspectRatio: false,
                scales: {
                    y: {
                        beginAtZero: true,
                        ticks: {
                            color: document.documentElement.classList.contains('dark') ? '#E5E7EB' :
'#374151'
                        }
                    },
                    x: {
                        ticks: {
                            color: document.documentElement.classList.contains('dark') ? '#E5E7EB' :
'#374151',
                            maxRotation: 45
                        }
                    }
                },
                plugins: {
                    legend: {
                        labels: {
                            color: document.documentElement.classList.contains('dark') ? '#E5E7EB' :
'#374151'
                        }
                    }
                }
            }
        });
    }

    function createSoilChart() {
        const ctx = document.getElementById('soilChart').getContext('2d');
```

```
        // Count soil types
        const soilTypes = {};
        cropsData.crops.forEach(crop => {
            const soil = crop.environmental_needs.soil;
            const key = soil.includes('loam') ? 'Loamy' :
                        soil.includes('clay') ? 'Clayey' :
                        soil.includes('sandy') ? 'Sandy' : 'Other';
            soilTypes[key] = (soilTypes[key] || 0) + 1;
        });

        ctx.chart = new Chart(ctx, {
            type: 'pie',
            data: {
                labels: Object.keys(soilTypes),
                datasets: [{
                    data: Object.values(soilTypes),
                    backgroundColor: ['#10B981', '#F59E0B', '#EF4444', '#8B5CF6'],
                    borderColor: ['#059669', '#D97706', '#DC2626', '#7C3AED'],
                    borderWidth: 2
                }]
            },
            options: {
                responsive: true,
                maintainAspectRatio: false,
                plugins: {
                    legend: {
                        position: 'bottom',
                        labels: {
                            color: document.documentElement.classList.contains('dark') ? '#E5E7EB' :
'#374151'
                        }
                    }
                }
            }
        });
    }

    // Compare functionality
    function initializeCompareSection() {
        const compareSelection = document.getElementById('compareSelection');
        compareSelection.innerHTML = cropsData.crops.map(crop => `
            <button class="compare-crop-btn text-left p-3 border border-gray-200 dark:border-
gray-600 rounded-lg hover:bg-gray-50 dark:hover:bg-gray-700 transition-colors ${
selectedCropsForComparison.includes(crop.name) ? 'bg-primary text-white border-primary' : ''}"
                    onclick="toggleCropSelection('${crop.name}')"
                    data-crop="${crop.name}">
                <div class="flex items-center space-x-2">
                    <span class="text-lg">${getCropEmoji(crop.name)}</span>
                    <span class="text-sm font-medium">${crop.name}</span>
                </div>
            </button>
        `).join('');
    }

    function toggleCropSelection(cropName) {
        const index = selectedCropsForComparison.indexOf(cropName);
        const button = document.querySelector(`[data-crop="${cropName}"]`);

        if (index > -1) {
            // Remove from selection
```

```
            selectedCropsForComparison.splice(index, 1);
            button.classList.remove('bg-primary', 'text-white', 'border-primary');
        } else {
            // Add to selection (max 3)
            if (selectedCropsForComparison.length < 3) {
                selectedCropsForComparison.push(cropName);
                button.classList.add('bg-primary', 'text-white', 'border-primary');
            } else {
                alert('You can compare maximum 3 crops at a time.');
                return;
            }
        }

        updateComparison();
    }

    function updateComparison() {
        const comparisonResults = document.getElementById('comparisonResults');
        const comparisonTable = document.getElementById('comparisonTable');

        if (selectedCropsForComparison.length < 2) {
            comparisonResults.classList.add('hidden');
            return;
        }

        const selectedCrops = selectedCropsForComparison.map(name =>
            cropsData.crops.find(crop => crop.name === name)
        );

        comparisonTable.innerHTML = `
            <div class="overflow-x-auto">
                <table class="w-full border-collapse">
                    <thead>
                        <tr class="bg-gray-50 dark:bg-gray-700">
                            <th class="border border-gray-200 dark:border-gray-600 px-4 py-3 text-left
font-medium text-gray-900 dark:text-white">Characteristic</th>
                            ${selectedCrops.map(crop => `
                                <th class="border border-gray-200 dark:border-gray-600 px-4 py-3 text-left
font-medium text-gray-900 dark:text-white">
                                    ${getCropEmoji(crop.name)} ${crop.name}
                                </th>
                            `).join('')}
                        </tr>
                    </thead>
                    <tbody class="divide-y divide-gray-200 dark:divide-gray-600">
                        <tr>
                            <td class="border border-gray-200 dark:border-gray-600 px-4 py-3 font-
medium text-gray-900 dark:text-white">Temperature</td>
                            ${selectedCrops.map(crop => `
                                <td class="border border-gray-200 dark:border-gray-600 px-4 py-3 text-
gray-700 dark:text-gray-300">${crop.environmental_needs.temp}</td>
                            `).join('')}
                        </tr>
                        <tr>
                            <td class="border border-gray-200 dark:border-gray-600 px-4 py-3 font-
medium text-gray-900 dark:text-white">Rainfall</td>
                            ${selectedCrops.map(crop => `
                                <td class="border border-gray-200 dark:border-gray-600 px-4 py-3 text-
gray-700 dark:text-gray-300">${crop.environmental_needs.rainfall}</td>
                            `).join('')}
```

```
                    </tr>
                    <tr>
                        <td class="border border-gray-200 dark:border-gray-600 px-4 py-3 font-
medium text-gray-900 dark:text-white">Sunlight</td>
                        ${selectedCrops.map(crop => `
                            <td class="border border-gray-200 dark:border-gray-600 px-4 py-3 text-
gray-700 dark:text-gray-300">${crop.environmental_needs.sunlight}</td>
                        `).join('')}
                    </tr>
                    <tr>
                        <td class="border border-gray-200 dark:border-gray-600 px-4 py-3 font-
medium text-gray-900 dark:text-white">Soil Type</td>
                        ${selectedCrops.map(crop => `
                            <td class="border border-gray-200 dark:border-gray-600 px-4 py-3 text-
gray-700 dark:text-gray-300">${crop.environmental_needs.soil}</td>
                        `).join('')}
                    </tr>
                    <tr>
                        <td class="border border-gray-200 dark:border-gray-600 px-4 py-3 font-
medium text-gray-900 dark:text-white">Humidity</td>
                        ${selectedCrops.map(crop => `
                            <td class="border border-gray-200 dark:border-gray-600 px-4 py-3 text-
gray-700 dark:text-gray-300">${crop.environmental_needs.humidity}</td>
                        `).join('')}
                    </tr>
                </tbody>
            </table>
        </div>
    `;

    comparisonResults.classList.remove('hidden');
}

// Get crop emoji based on name
function getCropEmoji(cropName) {
    const emojiMap = {
        'Wheat': '🌾',
        'Rice': '🍚',
        'Maize (Corn)': '🌽',
        'Sugarcane': '🎋',
        'Cotton': '☁️',
        'Barley': '🌾',
        'Groundnut (Peanut)': '🥜',
        'Soybean': '🫘',
        'Millet (e.g., Bajra)': '🌾',
        'Mustard': '💛',
        'Bambara Groundnut': '🥜',
        'Pearl Millet': '🌾',
        'Sorghum': '🌾',
        'Potato': '🥔',
        'Cassava': '🍠',
        'Sunflower': '🌻',
```

```
            'Sweet Potato': '🍠',
            'Carrot': '🥕',
            'Tomato': '🍅',
            'Banana': '🍌'
        };
        return emojiMap[cropName] || '🌱';
    }

    // Get rainfall category for color coding
    function getRainfallCategory(rainfallStr) {
        const match = rainfallStr.match(/(\d+)--(\d+)/);
        if (!match) return 'medium';
        const avg = (parseInt(match[1]) + parseInt(match[2])) / 2;
        if (avg < 50) return 'low';
        if (avg > 120) return 'high';
        return 'medium';
    }

    // Create crop card HTML
    function createCropCard(crop) {
        const emoji = getCropEmoji(crop.name);
        const rainfallCategory = getRainfallCategory(crop.environmental_needs.rainfall);
        const rainfallColorClass = {
            'low': 'bg-yellow-100 text-yellow-800 dark:bg-yellow-900 dark:text-yellow-200',
            'medium': 'bg-blue-100 text-blue-800 dark:bg-blue-900 dark:text-blue-200',
            'high': 'bg-blue-100 text-blue-800 dark:bg-blue-900 dark:text-blue-200'
        }[rainfallCategory];

        return `
            <div class="crop-card bg-white dark:bg-gray-800 rounded-lg shadow-md
hover:shadow-lg border border-gray-200 dark:border-gray-700 p-6 cursor-pointer"
onclick="showCropDetails('${crop.name}')">
                <div class="flex items-start justify-between mb-4">
                    <div class="flex items-center space-x-3">
                        <div class="text-3xl">${emoji}</div>
                        <h3 class="text-lg font-semibold text-gray-900 dark:text-white">${crop.name}</
h3>
                    </div>
                    <span class="px-2 py-1 text-xs font-medium rounded-full ${rainfallColorClass}">
                        ${crop.environmental_needs.rainfall}
                    </span>
                </div>

                <div class="space-y-3">
                    <div class="flex items-center space-x-2 text-sm text-gray-600 dark:text-
gray-300">
                        <span class="font-medium">🌡️ Temp:</span>
                        <span>${crop.environmental_needs.temp}</span>
                    </div>
                    <div class="flex items-center space-x-2 text-sm text-gray-600 dark:text-
gray-300">
                        <span class="font-medium">🌱 Soil:</span>
                        <span class="truncate">${crop.environmental_needs.soil}</span>
                    </div>
                    <div class="flex items-center space-x-2 text-sm text-gray-600 dark:text-
gray-300">
```

```
                    <span class="font-medium">☀️ Sun:</span>
                    <span>${crop.environmental_needs.sunlight}</span>
                  </div>
                </div>

                <div class="mt-4 pt-4 border-t border-gray-200 dark:border-gray-600">
                  <div class="flex flex-wrap gap-1">
                    ${crop.cropping_techniques.slice(0, 2).map(technique => {
                      const type = technique.split(':')[0];
                      return `<span class="px-2 py-1 text-xs bg-green-100 text-green-800 dark:bg-
green-900 dark:text-green-200 rounded">${type}</span>`;
                    }).join('')}
                    ${crop.cropping_techniques.length > 2 ? `<span class="px-2 py-1 text-xs bg-
gray-100 text-gray-600 dark:bg-gray-700 dark:text-gray-300 rounded">+$
{crop.cropping_techniques.length - 2} more</span>` : ''}
                  </div>
                </div>
              </div>
            `;
          }

          // Filter crops based on current filter and search term
          function filterCrops() {
            let filteredCrops = cropsData.crops;

            // Apply search filter
            if (searchTerm) {
              filteredCrops = filteredCrops.filter(crop =>
                crop.name.toLowerCase().includes(searchTerm.toLowerCase())
              );
            }

            // Apply category filter
            if (currentFilter !== 'all') {
              filteredCrops = filteredCrops.filter(crop => {
                switch (currentFilter) {
                  case 'low-water':
                    const rainfallMax = parseInt(crop.environmental_needs.rainfall.split('--')[1]);
                    return rainfallMax <= 75;
                  case 'drought-resistant':
                    return crop.environmental_needs.weather_factors.some(factor =>
                      factor.toLowerCase().includes('drought') ||
                      factor.toLowerCase().includes('dry')
                    ) || crop.environmental_needs.humidity === 'Low';
                  case 'intercropping':
                    return crop.cropping_techniques.some(technique =>
                      technique.toLowerCase().includes('intercropping')
                    );
                  case 'rotation':
                    return crop.cropping_techniques.some(technique =>
                      technique.toLowerCase().includes('rotation')
                    );
                  default:
                    return true;
                }
              });
            }

            return filteredCrops;
          }
```

```
        // Render crops grid
        function renderCrops() {
            const filteredCrops = filterCrops();
            const cropsGrid = document.getElementById('cropsGrid');
            const noResults = document.getElementById('noResults');
            const cropCount = document.getElementById('cropCount');

            if (filteredCrops.length === 0) {
                cropsGrid.innerHTML = '';
                noResults.classList.remove('hidden');
            } else {
                noResults.classList.add('hidden');
                cropsGrid.innerHTML = filteredCrops.map(createCropCard).join('');
            }

            cropCount.textContent = filteredCrops.length;
        }

        // Show crop details in modal
        function showCropDetails(cropName) {
            const crop = cropsData.crops.find(c => c.name === cropName);
            if (!crop) return;

            const modal = document.getElementById('cropModal');
            const modalTitle = document.getElementById('modalTitle');
            const modalContent = document.getElementById('modalContent');

            modalTitle.textContent = `${getCropEmoji(crop.name)} ${crop.name}`;

            modalContent.innerHTML = `
                <div class="grid md:grid-cols-2 gap-6">
                    <div class="space-y-4">
                        <h3 class="text-lg font-semibold text-gray-900 dark:text-white
mb-3">Environmental Requirements</h3>
                        <div class="space-y-3">
                            <div class="flex items-center space-x-3">
                                <span class="text-lg">🌡️</span>
                                <div>
                                    <span class="font-medium text-gray-700 dark:text-
gray-300">Temperature:</span>
                                    <span class="ml-2 text-gray-600 dark:text-gray-400">$
{crop.environmental_needs.temp}</span>
                                </div>
                            </div>
                            <div class="flex items-center space-x-3">
                                <span class="text-lg">💧 </span>
                                <div>
                                    <span class="font-medium text-gray-700 dark:text-gray-300">Rainfall:</
span>
                                    <span class="ml-2 text-gray-600 dark:text-gray-400">$
{crop.environmental_needs.rainfall}</span>
                                </div>
                            </div>
                            <div class="flex items-center space-x-3">
                                <span class="text-lg">☀️</span>
                                <div>
                                    <span class="font-medium text-gray-700 dark:text-gray-300">Sunlight:</
span>
```

```
                        <span class="ml-2 text-gray-600 dark:text-gray-400">${crop.environmental_needs.sunlight}</span>
                    </div>
                </div>
                <div class="flex items-center space-x-3">
                    <span class="text-lg">🌱</span>
                    <div>
                        <span class="font-medium text-gray-700 dark:text-gray-300">Soil:</span>
                        <span class="ml-2 text-gray-600 dark:text-gray-400">${crop.environmental_needs.soil}</span>
                    </div>
                </div>
                <div class="flex items-center space-x-3">
                    <span class="text-lg">💨</span>
                    <div>
                        <span class="font-medium text-gray-700 dark:text-gray-300">Humidity:</span>
                        <span class="ml-2 text-gray-600 dark:text-gray-400">${crop.environmental_needs.humidity}</span>
                    </div>
                </div>
            </div>

            <div class="mt-6">
                <h4 class="font-medium text-gray-700 dark:text-gray-300 mb-2">Key Weather Factors:</h4>
                <div class="flex flex-wrap gap-2">
                    ${crop.environmental_needs.weather_factors.map(factor =>
                        `<span class="px-2 py-1 text-xs bg-blue-100 text-blue-800 dark:bg-blue-900 dark:text-blue-200 rounded">${factor}</span>`
                    ).join('')}
                </div>
            </div>
        </div>

        <div class="space-y-4">
            <h3 class="text-lg font-semibold text-gray-900 dark:text-white mb-3">Cropping Techniques</h3>
            <div class="space-y-3">
                ${crop.cropping_techniques.map(technique => {
                    const [type, description] = technique.split(': ');
                    return `
                        <div class="border border-gray-200 dark:border-gray-600 rounded-lg p-3">
                            <h4 class="font-medium text-gray-800 dark:text-gray-200 mb-1">${type}</h4>
                            <p class="text-sm text-gray-600 dark:text-gray-400">${description || ''}</p>
                        </div>
                    `;
                }).join('')}
            </div>
        </div>
    </div>
`;

    modal.classList.remove('hidden');
    document.body.style.overflow = 'hidden';
}
```

```
        // Close crop details modal
        function closeCropModal() {
            const modal = document.getElementById('cropModal');
            modal.classList.add('hidden');
            document.body.style.overflow = 'auto';
        }

        // Event listeners
        document.getElementById('searchInput').addEventListener('input', (e) => {
            searchTerm = e.target.value;
            renderCrops();
        });

        document.querySelectorAll('.filter-chip').forEach(chip => {
            chip.addEventListener('click', (e) => {
                // Update active filter
                document.querySelectorAll('.filter-chip').forEach(c => {
                    c.classList.remove('bg-primary', 'text-white');
                    c.classList.add('bg-gray-100', 'dark:bg-gray-700', 'text-gray-700', 'dark:text-
gray-300');
                });

                e.target.classList.remove('bg-gray-100', 'dark:bg-gray-700', 'text-gray-700', 'dark:text-
gray-300');
                e.target.classList.add('bg-primary', 'text-white');

                currentFilter = e.target.dataset.filter;
                renderCrops();
            });
        });

        // Close modal when clicking outside
        document.getElementById('cropModal').addEventListener('click', (e) => {
            if (e.target.id === 'cropModal') {
                closeCropModal();
            }
        });

        // Close modal with escape key
        document.addEventListener('keydown', (e) => {
            if (e.key === 'Escape') {
                closeCropModal();
            }
        });

        // Initial setup
        document.addEventListener('DOMContentLoaded', () => {
            renderCrops();
            setupImageUpload();

            // Set initial active filter
            document.querySelector('[data-filter="all"]').classList.remove('bg-gray-100', 'dark:bg-
gray-700', 'text-gray-700', 'dark:text-gray-300');
            document.querySelector('[data-filter="all"]').classList.add('bg-primary', 'text-white');
        });
    </script>
</body>
</html>
```