

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Kisan AI - Smart Farming Assistant</title>
<script src="https://unpkg.com/react@18/umd/react.development.js"></script>
<script src="https://unpkg.com/react-dom@18/umd/react-dom.development.js"></script>
<script src="https://unpkg.com/@babel/standalone/babel.min.js"></script>
<link href="https://cdn.jsdelivr.net/npm/tailwindcss@2.2.19/dist/tailwind.min.css"
rel="stylesheet">
<link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.0/css/all.min.css">
<link
href="https://fonts.googleapis.com/css2?family=Inter:wght@300;400;500;600;700&family=O
pen+Sans:wght@400;600;700&display=swap" rel="stylesheet">
<style>
:root {
  --primary-color: #2F855A;
  --secondary-color: #38A169;
  --accent-color: #4299E1;
  --light-bg: #F7FAFC;
  --dark-text: #1A202C;
  --light-text: #718096;
}

body {
  font-family: 'Open Sans', sans-serif;
  background-color: #F8F9FA;
  color: var(--dark-text);
  line-height: 1.6;
}

.professional-typography {
  font-family: 'Inter', sans-serif;
}

.btn-primary {
  background-color: var(--primary-color);
  color: white;
  transition: all 0.3s ease;
}

.btn-primary:hover {
  background-color: #276749;
  transform: translateY(-2px);
  box-shadow: 0 4px 12px rgba(47, 133, 90, 0.2);
}
```

```
.btn-secondary {
  background-color: var(--accent-color);
  color: white;
  transition: all 0.3s ease;
}

.btn-secondary:hover {
  background-color: #3182CE;
  transform: translateY(-2px);
  box-shadow: 0 4px 12px rgba(66, 153, 225, 0.2);
}

.card {
  background: white;
  border-radius: 12px;
  box-shadow: 0 4px 6px rgba(0, 0, 0, 0.05);
  transition: all 0.3s ease;
}

.card:hover {
  transform: translateY(-5px);
  box-shadow: 0 10px 25px rgba(0, 0, 0, 0.1);
}

.feature-card {
  border-left: 4px solid var(--primary-color);
}

.nav-item {
  position: relative;
  transition: all 0.3s ease;
}

.nav-item:hover {
  color: var(--primary-color);
}

.nav-item.active {
  color: var(--primary-color);
  font-weight: 600;
}

.nav-item.active::after {
  content: "";
  position: absolute;
  bottom: -8px;
  left: 0;
```

```

width: 100%;
height: 3px;
background-color: var(--primary-color);
border-radius: 3px;
}

.hero-gradient {
  background: linear-gradient(135deg, rgba(47, 133, 90, 0.1) 0%, rgba(56, 161, 105, 0.1)
100%);
}

.voice-wave {
  animation: voice-wave 0.8s ease-in-out infinite alternate;
}

@keyframes voice-wave {
  0% { height: 8px; opacity: 0.4; }
  100% { height: 24px; opacity: 1;}
}

.loading-spinner {
  animation: spin 1s linear infinite;
}

@keyframes spin {
  0% { transform: rotate(0deg); }
  100% { transform: rotate(360deg); }
}

.slide-in {
  animation: slide-in 0.6s ease-out forwards;
}

@keyframes slide-in {
  from {
    opacity: 0;
    transform: translateY(20px);
  }
  to {
    opacity: 1;
    transform: translateY(0);
  }
}

.pulse {
  animation: pulse 2s infinite;
}

```

```
@keyframes pulse {
  0%, 100% { opacity: 1;}
  50% { opacity: 0.5;}
}

.input-field {
  transition: all 0.3s ease;
  border: 1px solid #E2E8F0;
}

.input-field:focus {
  border-color: var(--primary-color);
  box-shadow: 0 0 0 3px rgba(47, 133, 90, 0.2);
}

.badge {
  font-size: 0.75rem;
  padding: 0.25rem 0.5rem;
  border-radius: 9999px;
}

.badge-new {
  background-color: #F6E05E;
  color: #744210;
}

.badge-live {
  background-color: #F56565;
  color: white;
}

.badge-beta {
  background-color: #4299E1;
  color: white;
}

.tab-button {
  transition: all 0.3s ease;
  padding: 7px 15px;
  border-radius: 7px;
}

.tab-button.active {
  background-color: var(--primary-color);
  color: white;
  border-radius: 7px;
  padding: 7px 15px;
}
```

```
.tab-button:not(.active):hover {  
    background-color: #EDF2F7;  
}
```

```
.progress-bar {  
    transition: width 0.6s ease;  
}
```

```
/* Full-page view styles */
```

```
.full-page-view {  
    position: fixed;  
    top: 0;  
    left: 0;  
    right: 0;  
    bottom: 0;  
    background-color: white;  
    z-index: 1000;  
    overflow-y: auto;  
    padding: 20px;  
}
```

```
.back-button {  
    display: flex;  
    align-items: center;  
    color: var(--primary-color);  
    margin-bottom: 20px;  
    cursor: pointer;  
}
```

```
.back-button i {  
    margin-right: 8px;  
}
```

```
/* Login page styles */
```

```
.login-bg {  
    background: linear-gradient(rgba(0, 0, 0, 0.5), rgba(0, 0, 0, 0.5)),  
    url('https://images.unsplash.com/photo-1500382017468-9049fed747ef?ixlib=rb-4.0.3&ixid=M3wxMjA3fDB8MHxwaG90by1wYWdlfHx8fGVufDB8fHx8fA%3D%3D&auto=format&fit=crop&w=2070&q=80');  
    background-size: cover;  
    background-position: center;  
    min-height: 100vh;  
}
```

```
.login-card {  
    backdrop-filter: blur(10px);  
    background: rgba(255, 255, 255, 0.85);
```

```

}

.feature-icon {
  width: 60px;
  height: 60px;
  border-radius: 16px;
  display: flex;
  align-items: center;
  justify-content: center;
  font-size: 24px;
  margin-bottom: 16px;
}

/* Enhanced container styles */
.app-container {
  max-width: 1200px;
  margin: 0 auto;
  padding: 20px;
  background-color: rgba(255, 255, 255, 0);
  border-radius: 16px;
  box-shadow: 0 10px 30px rgba(0, 0, 0, 0.1);
  transition: all 0.3s ease;
}

.app-container:hover {
  box-shadow: 0 15px 35px rgba(0, 0, 0, 0.15);
}

/* Google button fix */
.google-btn img {
  width: 35px;
  height: 35px;
  margin-right: 8px;
}

.text-gray-500 {
  --tw-text-opacity: 1;
  color: rgb(47 133 90);
  padding-bottom: 7px;
  padding-top: 7px;
  padding-left: 15px;
  padding-right: 15px;
  border-radius: 7px;
}

/* Alert section scrolling */
.alert-section {
  max-height: 400px;

```

```

    overflow-y: auto;
    scroll-behavior: smooth;
}

/* Smooth scrolling for alerts */
.alert-section::-webkit-scrollbar {
    width: 8px;
}

.alert-section::-webkit-scrollbar-track {
    background: #ffffff;
    border-radius: 10px;
}

.alert-section::-webkit-scrollbar-thumb {
    background: var(--primary-color);
    border-radius: 10px;
}

.alert-section::-webkit-scrollbar-thumb:hover {
    background: #276749;
}

/* Weather alert styles */
.weather-alert {
    background-color: rgba(66, 153, 225, 0.1);
    border-left: 4px solid var(--accent-color);
}

/* Daily follow-up styles */
.followup-card {
    border-left: 4px solid var(--secondary-color);
}

.followup-progress {
    height: 8px;
    background-color: #E2E8F0;
    border-radius: 4px;
    overflow: hidden;
}

.followup-progress-bar {
    height: 100%;
    background-color: var(--secondary-color);
    transition: width 0.6s ease;
}

/* Image upload preview */

```

```
.upload-preview {  
  position: relative;  
}
```

```
.upload-preview img {  
  max-height: 300px;  
  object-fit: contain;  
}
```

```
.upload-progress {  
  position: absolute;  
  bottom: 0;  
  left: 0;  
  right: 0;  
  height: 4px;  
  background-color: rgba(255, 255, 255, 0.5);  
}
```

```
.upload-progress-bar {  
  height: 100%;  
  background-color: var(--primary-color);  
  transition: width 0.3s ease;  
}
```

/ Crop timeline */*

```
.timeline {  
  position: relative;  
  padding-left: 30px;  
}
```

```
.timeline::before {  
  content: "";  
  position: absolute;  
  left: 10px;  
  top: 0;  
  bottom: 0;  
  width: 2px;  
  background-color: #E2E8F0;  
}
```

```
.timeline-item {  
  position: relative;  
  padding-bottom: 20px;  
}
```

```
.timeline-item::before {  
  content: "";  
  position: absolute;
```



```
    left: -30px;
    top: 0;
    width: 20px;
    height: 20px;
    border-radius: 50%;
    background-color: var(--primary-color);
    border: 4px solid white;
}
```

```
.timeline-date {
    font-weight: 600;
    color: var(--primary-color);
}
```

```
.timeline-content {
    background-color: #F7FAFC;
    border-radius: 8px;
    padding: 12px;
    margin-top: 8px;
}
```

```
/* Password toggle */
.password-toggle {
    position: absolute;
    right: 10px;
    top: 50%;
    transform: translateY(-50%);
    cursor: pointer;
    color: var(--light-text);
}
```

```
.password-toggle:hover {
    color: var(--primary-color);
}
```

```
/* Empty state */
.empty-state {
    text-align: center;
    padding: 40px 20px;
}
```

```
.empty-state-icon {
    width: 80px;
    height: 80px;
    margin: 0 auto 20px;
    display: flex;
    align-items: center;
    justify-content: center;
```

```

    background-color: #F7FAFC;
    border-radius: 50%;
    color: var(--light-text);
    font-size: 32px;
  }

  /* Active follow-up indicator */
  .active-followup {
    position: relative;
  }

  .active-followup::after {
    content: "";
    position: absolute;
    right: -10px;
    top: 50%;
    transform: translateY(-50%);
    width: 8px;
    height: 8px;
    background-color: var(--primary-color);
    border-radius: 50%;
    animation: pulse 2s infinite;
  }
</style>
</head>
<body class="bg-gray-50">
<div id="root"></div>

<script type="text/babel">
  const { useState, useEffect, useRef } = React;

  // Enhanced mock data with more professional structure
  const mockCrops = [
    {
      name: 'Basmati Rice',
      price: '$3,250',
      unit: 'per quintal',
      change: '+8.5%',
      trend: 'up',
      volume: '2,340 tons',
      forecast: 'bullish',
      quality: 'Grade A',
      image: 'rice.png'
    },
    {
      name: 'Wheat (PBW 343)',
      price: '$2,850',
      unit: 'per quintal',

```

```

      change: '-2.3%',
      trend: 'down',
      volume: '1,890 tons',
      forecast: 'stable',
      quality: 'Premium',
      image: 'wheat.png'
    },
    {
      name: 'Hybrid Corn',
      price: '$2,100',
      unit: 'per quintal',
      change: '+12.8%',
      trend: 'up',
      volume: '3,120 tons',
      forecast: 'bullish',
      quality: 'Grade A',
      image: 'corn.png'
    },
    {
      name: 'Tomato (Fresh)',
      price: '$65',
      unit: 'per kg',
      change: '+25.4%',
      trend: 'up',
      volume: '890 tons',
      forecast: 'volatile',
      quality: 'Grade A',
      image: 'tomato.png'
    }
  ];

```

```

const mockSchemes = [
  {
    name: 'PM-KISAN Samman Nidhi',
    amount: '$6,000',
    period: 'per year',
    type: 'Central',
    beneficiaries: '11.7 Cr',
    status: 'Active',
    icon: 'hand-holding-usd',
    description: 'Provides income support to all landholding farmers to enable them to take care of expenses related to agriculture and allied activities as well as domestic needs.',
    eligibility: 'All landholding farmers families, Small and marginal farmers families, Cultivators',
    documents: 'Aadhaar Card, Land ownership documents, Bank account details'
  },
  {
    name: 'Pradhan Mantri Fasal Bima Yojana',

```

```

    amount: '$2,00,000',
    period: 'coverage',
    type: 'Central',
    beneficiaries: '5.5 Cr',
    status: 'Active',
    icon: 'shield-alt',
    description: 'Provides comprehensive insurance cover against failure of the crop thus
helping in stabilising the income of the farmers.',
    eligibility: 'All farmers including sharecroppers and tenant farmers growing notified
crops in notified areas',
    documents: 'Land records, Aadhaar Card, Bank account details'
  },
  {
    name: 'Kisan Credit Card',
    amount: '$3,00,000',
    period: 'limit',
    type: 'Central',
    beneficiaries: '7.3 Cr',
    status: 'Active',
    icon: 'credit-card',
    description: 'Provides adequate and timely credit support from the banking system to
farmers for their cultivation needs.',
    eligibility: 'Individual farmers, Joint borrowers, Tenant farmers, Oral lessees,
Sharecroppers',
    documents: 'Land ownership documents, Identity proof, Address proof, Passport size
photos'
  },
  {
    name: 'Soil Health Card Scheme',
    amount: 'Free Testing',
    period: 'service',
    type: 'Central',
    beneficiaries: '22 Cr',
    status: 'Active',
    icon: 'flask',
    description: 'Provides information to farmers on nutrient status of their soil along with
recommendations on appropriate dosage of nutrients to be applied for improving soil health
and its fertility.',
    eligibility: 'All farmers across the country.',
    documents: 'Aadhaar Card, Land ownership documents'
  }
];

```

// Enhanced voice recognition hook

```

const useVoiceRecognition = () => {
  const [isListening, setIsListening] = useState(false);
  const [transcript, setTranscript] = useState("");
  const [confidence, setConfidence] = useState(0);

```

```

const recognition = useRef(null);

useEffect(() => {
  if ('webkitSpeechRecognition' in window) {
    recognition.current = new window.webkitSpeechRecognition();
    recognition.current.continuous = true;
    recognition.current.interimResults = true;
    recognition.current.lang = 'hi-IN';
    recognition.current.onresult = (event) => {
      const transcript = Array.from(event.results)
        .map(result => result[0])
        .map(result => result.transcript)
        .join("");
      setTranscript(transcript);
      if (event.results[event.results.length - 1].isFinal) {
        setConfidence(event.results[event.results.length - 1][0].confidence);
      }
    };
    recognition.current.onend = () => {
      setIsListening(false);
    };
  }
}, []);

const startListening = () => {
  if (recognition.current) {
    setIsListening(true);
    recognition.current.start();
  }
};

const stopListening = () => {
  if (recognition.current) {
    recognition.current.stop();
    setIsListening(false);
  }
};

return { isListening, transcript, confidence, startListening, stopListening };
};

```

// Login Page Component

```

const LoginPage = ({ onLoginSuccess }) => {
  const [isLogin, setIsLogin] = useState(true);
  const [email, setEmail] = useState("");
  const [password, setPassword] = useState("");
  const [name, setName] = useState("");
  const [phone, setPhone] = useState("");

```

```

const [errors, setErrors] = useState({});
const [showPassword, setShowPassword] = useState(false);

// Dummy credentials for demo
const DEMO_EMAIL = "farmer@demo.com";
const DEMO_PASSWORD = "kisan123";

const validateLogin = () => {
  const newErrors = {};
  if (!email) newErrors.email = 'Email is required';
  if (!password) newErrors.password = 'Password is required';
  setErrors(newErrors);
  return Object.keys(newErrors).length === 0;
};

const validateRegister = () => {
  const newErrors = {};
  if (!name) newErrors.name = 'Name is required';
  if (!email) newErrors.email = 'Email is required';
  if (!phone) newErrors.phone = 'Phone is required';
  if (!password) newErrors.password = 'Password is required';
  setErrors(newErrors);
  return Object.keys(newErrors).length === 0;
};

const handleLogin = (e) => {
  e.preventDefault();
  if (validateLogin()) {
    // For demo purposes, accept any credentials but show success only for demo
    credentials
    if (email === DEMO_EMAIL && password === DEMO_PASSWORD) {
      onLoginSuccess({
        name: "Demo Farmer",
        email: DEMO_EMAIL,
        phone: "9876543210"
      });
    } else {
      alert("Invalid credentials. For demo, use:\nEmail:
farmer@demo.com\nPassword: kisan123");
    }
  }
};

const handleRegister = (e) => {
  e.preventDefault();
  if (validateRegister()) {
    // Simulate registration
    onLoginSuccess({

```

```

        name,
        email,
        phone
    });
}
};

const handleGoogleLogin = () => {
    // Simulate Google login
    onLoginSuccess({
        name: "Google User",
        email: "googleuser@gmail.com",
        phone: "9876543210"
    });
};

return (
    <div className="login-bg flex items-center justify-center p-4">
        <div className="app-container max-w-5xl w-full grid grid-cols-1 lg:grid-cols-2
gap-8">
            {/* Left side - App Info */}
            <div className="text-white">
                <div className="flex items-center space-x-3 mb-6">
                    <div className="w-12 h-12 bg-white bg-opacity-20 rounded-lg flex
items-center justify-center">
                        <i className="fas fa-seedling text-white text-xl"></i>
                    </div>
                    <h1 className="text-3xl font-bold">Kisan AI</h1>
                </div>
                <h2 className="text-4xl font-bold mb-6">Your Smart Farming
Companion</h2>
                <p className="text-xl mb-8">AI-powered agricultural assistant to maximize
your crop yield and profits</p>
                <div className="grid grid-cols-2 gap-6">
                    <div className="bg-white bg-opacity-10 p-6 rounded-xl">
                        <div className="feature-icon bg-green-100 text-green-600">
                            <i className="fas fa-leaf"></i>
                        </div>
                        <h3 className="text-xl font-semibold mb-2">Crop Doctor</h3>
                        <p className="text-white text-opacity-80">Diagnose plant diseases
with AI</p>
                    </div>
                    <div className="bg-white bg-opacity-10 p-6 rounded-xl">
                        <div className="feature-icon bg-blue-100 text-blue-600">
                            <i className="fas fa-chart-line"></i>
                        </div>
                        <h3 className="text-xl font-semibold mb-2">Market Prices</h3>
                        <p className="text-white text-opacity-80">Real time mandi prices</p>
                    </div>
                </div>
            </div>

```

```

    </div>
    <div className="bg-white bg-opacity-10 p-6 rounded-xl">
      <div className="feature-icon bg-purple-100 text-purple-600">
        <i className="fas fa-microphone"></i>
      </div>
      <h3 className="text-xl font-semibold mb-2">Voice Assistant</h3>
      <p className="text-white text-opacity-80">Ask in your local
language</p>
    </div>
    <div className="bg-white bg-opacity-10 p-6 rounded-xl">
      <div className="feature-icon bg-yellow-100 text-yellow-600">
        <i className="fas fa-seedling"></i>
      </div>
      <h3 className="text-xl font-semibold mb-2">Crop Advisor</h3>
      <p className="text-white text-opacity-80">Personalized
recommendations</p>
    </div>
  </div>

  { /* Right side - Login/Register Form */ }
  <div className="login-card rounded-2xl shadow-xl p-8">
    <div className="text-center mb-8">
      <h2 className="text-2xl font-bold text-gray-800">
        {isLogin ? 'Welcome Back' : 'Create Account'}
      </h2>
      <p className="text-gray-600">
        {isLogin ? 'Sign in to continue to Kisan AI' : 'Join Kisan AI to access
smart farming tools'}
      </p>
    </div>

    <button
      onClick={handleGoogleLogin}
      className="google-btn w-full flex items-center justify-center space-x-3
bg-white border border-gray-300 rounded-lg py-3 px-4 hover:bg-gray-50 transition mb-4"
    >
      
      <span>{isLogin ? 'Sign in with Google' : 'Sign up with Google'}</span>
    </button>

    <div className="relative my-6">
      <div className="absolute inset-0 flex items-center">
        <div className="w-full border-t border-gray-300"></div>
      </div>
      <div className="relative flex justify-center text-sm">

```



```

        <span className="px-2 bg-white text-gray-500">Or continue with
email</span>
    </div>
</div>

{isLogin ? (
    <form onSubmit={handleLogin} className="space-y-4">
        <div>
            <label htmlFor="email" className="block text-sm font-medium
text-gray-700 mb-1">Email</label>
            <input
                type="email"
                id="email"
                value={email}
                onChange={(e) => setEmail(e.target.value)}
                className={`w-full input-field py-2 px-3 rounded-lg ${errors.email ?
'border-red-500' : ''}`}
                placeholder="farmer@example.com"
            />
            {errors.email && <p className="text-red-500 text-xs
mt-1">{errors.email}</p>}
        </div>
        <div className="relative">
            <label htmlFor="password" className="block text-sm font-medium
text-gray-700 mb-1">Password</label>
            <input
                type={showPassword ? "text" : "password"}
                id="password"
                value={password}
                onChange={(e) => setPassword(e.target.value)}
                className={`w-full input-field py-2 px-3 rounded-lg
${errors.password ? 'border-red-500' : ''}`}
                placeholder="*****"
            />
            <span
                className="password-toggle"
                onClick={() => setShowPassword(!showPassword)}
            >
                <i className={`fas ${showPassword ? 'fa-eye-slash' :
'fa-eye'}}`></i>
            </span>
            {errors.password && <p className="text-red-500 text-xs
mt-1">{errors.password}</p>}
        </div>
        <div className="flex items-center justify-between">
            <div className="flex items-center">
                <input
                    id="remember-me"

```

```

        name="remember-me"
        type="checkbox"
        className="h-4 w-4 text-green-600 focus:ring-green-600
border-gray-300 rounded"
      />
      <label htmlFor="remember-me" className="ml-2 block text-sm
text-gray-700">
        Remember me
      </label>
    </div>
    <a href="#" className="text-sm text-green-600
hover:text-green-500">
      Forgot password?
    </a>
  </div>
  <button type="submit" className="btn-primary w-full py-3 px-4
rounded-lg">
    Sign In
  </button>
  <div className="text-center text-sm text-gray-600">
    Don't have an account?{' '}
    <button
      type="button"
      onClick={() => setIsLogin(false)}
      className="text-green-600 font-medium ml-1"
    >
      Sign Up
    </button>
  </div>
</form>
): (
  <form onSubmit={handleRegister} className="space-y-4">
    <div>
      <label htmlFor="name" className="block text-sm font-medium
text-gray-700 mb-1">Full Name</label>
      <input
        type="text"
        id="name"
        value={name}
        onChange={(e) => setName(e.target.value)}
        className={`w-full input-field py-2 px-3 rounded-lg ${errors.name
? 'border-red-500' : ''}`}
        placeholder="Farmer Name"
      />
      {errors.name && <p className="text-red-500 text-xs
mt-1">{errors.name}</p>}
    </div>
  </div>

```

```

        <label htmlFor="email" className="block text-sm font-medium
text-gray-700 mb-1">Email</label>
        <input
            type="email"
            id="email"
            value={email}
            onChange={(e) => setEmail(e.target.value)}
            className={`w-full input-field py-2 px-3 rounded-lg ${errors.email ?
'border-red-500' : ''}`}
            placeholder="farmer@example.com"
        />
        {errors.email && <p className="text-red-500 text-xs
mt-1">{errors.email}</p>}
    </div>
</div>
    <label htmlFor="phone" className="block text-sm font-medium
text-gray-700 mb-1">Phone Number</label>
    <input
        type="tel"
        id="phone"
        value={phone}
        onChange={(e) => setPhone(e.target.value)}
        className={`w-full input-field py-2 px-3 rounded-lg ${errors.phone
? 'border-red-500' : ''}`}
        placeholder="+91 9876543210"
    />
    {errors.phone && <p className="text-red-500 text-xs
mt-1">{errors.phone}</p>}
</div>
<div className="relative">
    <label htmlFor="password" className="block text-sm font-medium
text-gray-700 mb-1">Password</label>
    <input
        type={showPassword ? "text" : "password"}
        id="password"
        value={password}
        onChange={(e) => setPassword(e.target.value)}
        className={`w-full input-field py-2 px-3 rounded-lg
${errors.password ? 'border-red-500' : ''}`}
        placeholder="*****"
    />
    <span
        className="password-toggle"
        onClick={() => setShowPassword(!showPassword)}
    >
        <i className={`fas ${showPassword ? 'fa-eye-slash' :
'fa-eye'}}`></i>
    </span>

```

```

        {errors.password && <p className="text-red-500 text-xs
mt-1">{errors.password}</p>}
      </div>
      <button type="submit" className="btn-primary w-full py-3 px-4
rounded-lg">
        Create Account
      </button>
      <div className="text-center text-sm text-gray-600">
        Already have an account?{' '}
        <button
          type="button"
          onClick={() => setIsLogin(true)}
          className="text-green-600 font-medium ml-1"
        >
          Sign In
        </button>
      </div>
    </form>
  )}
</div>
</div>
</div>
);
};

```

// Professional Navigation Component

```

const Navigation = ({ activeTab, setActiveTab, onLogout }) => {
  const [isScrolled, setIsScrolled] = useState(false);

```

```

  useEffect(() => {
    const handleScroll = () => {
      setIsScrolled(window.scrollY > 20);
    };
    window.addEventListener('scroll', handleScroll);
    return () => window.removeEventListener('scroll', handleScroll);
  }, []);

```

```

const tabs = [
  { id: 'dashboard', label: 'Dashboard', icon: 'chart-pie', badge: null },
  { id: 'crop-disease', label: 'Crop Doctor', icon: 'leaf', badge: 'AI' },
  { id: 'market', label: 'Market Prices', icon: 'chart-line', badge: 'Live' },
  { id: 'voice', label: 'Voice Assistant', icon: 'microphone', badge: 'Beta' },
  { id: 'recommendations', label: 'Crop Advisor', icon: 'seedling', badge: null },
  { id: 'schemes', label: 'Govt Schemes', icon: 'file-invoice-dollar', badge: null }
];

```

```

return (

```

```

        <header className={`sticky top-0 z-50 bg-white shadow-sm transition-all
duration-300 ${isScrolled ? 'py-2' : 'py-4'}}`>
        <div className="container mx-auto px-4">
        <div className="flex items-center justify-between">
        <div className="flex items-center space-x-2">
        <div className="w-10 h-10 bg-green-600 rounded-lg flex items-center
justify-center">
        <i className="fas fa-seedling text-white"></i>
        </div>
        <h1 className="text-xl font-bold text-gray-800">Kisan AI</h1>
        </div>

        <nav className="hidden md:flex items-center space-x-8">
        {tabs.map((tab) => (
        <button
        key={tab.id}
        onClick={() => setActiveTab(tab.id)}
        className={`nav-item ${activeTab === tab.id ? 'active' : ''}`}
        >
        <i className={`fas fa-${tab.icon} mr-2`} ></i>
        {tab.label}
        {tab.badge && (
        <span className={`badge badge-${tab.badge.toLowerCase()}
ml-2`} >
        {tab.badge}
        </span>
        ))}
        </button>
        )))}
        </nav>

        <div className="flex items-center space-x-4">
        <button
        onClick={onLogout}
        className="hidden md:block bg-gray-100 text-gray-800 px-4 py-2
rounded-lg hover:bg-gray-200 transition"
        >
        <i className="fas fa-sign-out-alt mr-2"></i>
        Logout
        </button>
        <button className="md:hidden text-gray-600">
        <i className="fas fa-bars text-xl"></i>
        </button>
        </div>
        </div>
        </div>
        </header>
    );

```

```
};
```

```
// Enhanced Dashboard Component
```

```
const Dashboard = ({ setActiveTab }) => {
```

```
  const [currentTime, setCurrentTime] = useState(new Date());
```

```
  const [weatherData, setWeatherData] = useState({
```

```
    temp: 28,
```

```
    condition: 'sunny',
```

```
    humidity: 65,
```

```
    windSpeed: 12,
```

```
    forecast: [
```

```
      { day: 'Today', high: 28, low: 22, condition: 'sunny', rain: 0 },
```

```
      { day: 'Tomorrow', high: 27, low: 21, condition: 'partly-cloudy', rain: 10 },
```

```
      { day: 'Wed', high: 26, low: 20, condition: 'cloudy', rain: 30 },
```

```
      { day: 'Thu', high: 25, low: 19, condition: 'rain', rain: 70 },
```

```
      { day: 'Fri', high: 24, low: 18, condition: 'rain', rain: 80 }
```

```
    ]
```

```
  });
```

```
  const [reminders, setReminders] = useState([]);
```

```
  const [weatherAlerts, setWeatherAlerts] = useState([]);
```

```
  const [showWeatherAlertForm, setShowWeatherAlertForm] = useState(false);
```

```
  const [weatherAlertCondition, setWeatherAlertCondition] = useState('rain');
```

```
  const [weatherAlertValue, setWeatherAlertValue] = useState('50');
```

```
  const [weatherAlertUnit, setWeatherAlertUnit] = useState('mm');
```

```
  const [activeFollowUp, setActiveFollowUp] = useState(null);
```

```
  const weatherAlertRef = useRef(null);
```

```
  useEffect(() => {
```

```
    const timer = setInterval(() => setCurrentTime(new Date()), 1000);
```

```
    return () => clearInterval(timer);
```

```
  }, []);
```

```
  const handleAddReminder = (reminderText) => {
```

```
    if (reminderText.trim()) {
```

```
      setReminders([...reminders, {
```

```
        id: Date.now(),
```

```
        text: reminderText,
```

```
        date: new Date().toLocaleDateString()
```

```
      }]);
```

```
    }
```

```
  };
```

```
  const handleAddWeatherAlert = () => {
```

```
    const newAlert = {
```

```
      id: Date.now(),
```

```
      condition: weatherAlertCondition,
```

```
      value: weatherAlertValue,
```

```
      unit: weatherAlertUnit
```

```

    };
    setWeatherAlerts([...weatherAlerts, newAlert]);
    setShowWeatherAlertForm(false);

    // Scroll to the alerts section after adding
    setTimeout(() => {
      weatherAlertRef.current?.scrollIntoView({ behavior: 'smooth' });
    }, 100);
  };

  const handleRemoveWeatherAlert = (id) => {
    setWeatherAlerts(weatherAlerts.filter(alert => alert.id !== id));
  };

  const handleEditWeatherAlert = (alert) => {
    setWeatherAlertCondition(alert.condition);
    setWeatherAlertValue(alert.value);
    setWeatherAlertUnit(alert.unit);
    setShowWeatherAlertForm(true);
    handleRemoveWeatherAlert(alert.id);

    // Scroll to the form after clicking edit
    setTimeout(() => {
      weatherAlertRef.current?.scrollIntoView({ behavior: 'smooth' });
    }, 100);
  };

  // Simulate active follow-up (would come from backend in real app)
  useEffect(() => {
    // This would normally come from an API call
    const mockActiveFollowUp = {
      crop: 'Wheat',
      day: 45,
      task: 'Second weeding and irrigation',
      progress: 60,
      nextTask: 'Monitor for diseases and pests (Day 60)'
    };
    setActiveFollowUp(mockActiveFollowUp);
  }, []);

  return (
    <div className="container mx-auto px-4 py-8 space-y-8 slide-in">
      { /* Hero Section */ }
      <div className="hero-gradient rounded-2xl p-8 text-center">
        <h1 className="text-3xl md:text-4xl font-bold text-gray-800 mb-4">
          Welcome to Your Smart Farming Assistant
        </h1>
        <p className="text-lg text-gray-600 max-w-2xl mx-auto">

```

Get AI-powered insights to maximize your crop yield and profits

```
<div className="flex flex-wrap items-center justify-center gap-4 mt-6">
  <div className="flex items-center bg-white px-4 py-2 rounded-lg
shadow-sm">
    <i className="fas fa-map-marker-alt text-green-600 mr-2"></i>
    <span>Tirupati, Andhra Pradesh</span>
  </div>
  <div className="flex items-center bg-white px-4 py-2 rounded-lg
shadow-sm">
    <i className="fas fa-clock text-green-600 mr-2"></i>
    <span>{currentTime.toLocaleTimeString()}</span>
  </div>
  <div className="flex items-center bg-white px-4 py-2 rounded-lg
shadow-sm">
    <i className="fas fa-calendar-day text-green-600 mr-2"></i>
    <span>{currentTime.toLocaleDateString()}</span>
  </div>
</div>

{/* Active Follow-up Section */}
{activeFollowUp && (
  <div className="bg-white rounded-2xl shadow-sm p-6 followup-card">
    <div className="flex items-center justify-between mb-4">
      <h3 className="text-xl font-semibold text-gray-800 flex items-center">
        <i className="fas fa-calendar-check text-green-500 mr-3"></i>
        Active Daily Follow-up
      </h3>
      <span className="text-sm text-green-600">Day
{activeFollowUp.day}</span>
    </div>

    <div className="mb-4">
      <div className="flex justify-between text-sm mb-1">
        <span>Progress</span>
        <span>{activeFollowUp.progress}%</span>
      </div>
      <div className="w-full bg-gray-200 rounded-full h-2">
        <div
          className="bg-green-500 h-2 rounded-full"
          style={{ width: `${activeFollowUp.progress}%` }}
        ></div>
      </div>
    </div>

    <div className="space-y-3">
```



```

        <div className="bg-green-50 rounded-lg p-4">
          <h4 className="text-green-800 font-medium mb-2">Current Task</h4>
          <p>{activeFollowUp.task}</p>
        </div>

        <div className="bg-blue-50 rounded-lg p-4">
          <h4 className="text-blue-800 font-medium mb-2">Next Task</h4>
          <p>{activeFollowUp.nextTask}</p>
        </div>
      </div>

      <div className="mt-4 pt-4 border-t border-gray-100">
        <button
          onClick={() => setActiveTab('recommendations')}
          className="btn-primary w-full py-2 px-4 rounded-lg"
        >
          <i className="fas fa-tasks mr-2"></i>
          View All Follow-ups
        </button>
      </div>
    </div>
  )}

```

```

  { /* Feature Cards */ }

```

```

  <div className="grid grid-cols-1 md:grid-cols-2 lg:grid-cols-4 gap-6">
    {[
      {
        title: 'Crop Doctor',
        desc: 'Diagnose plant diseases with AI',
        icon: 'leaf',
        color: 'text-green-500',
        stats: '95% Accuracy',
        onClick: () => setActiveTab('crop-disease')
      },
      {
        title: 'Market Prices',
        desc: 'Real-time prices from 500+ markets',
        icon: 'chart-line',
        color: 'text-blue-500',
        stats: 'Live Updates',
        onClick: () => setActiveTab('market')
      },
      {
        title: 'Voice Assistant',
        desc: 'Ask questions in your language',
        icon: 'microphone',
        color: 'text-purple-500',
        stats: '12 Languages',

```

```

        onClick: () => setActiveTab('voice')
      },
    {
      title: 'Crop Advisor',
      desc: 'Personalized planting advice',
      icon: 'seedling',
      color: 'text-yellow-500',
      stats: 'ML Powered',
      onClick: () => setActiveTab('recommendations')
    }
  ].map((card, index) => (
    <button
      key={index}
      onClick={card.onClick}
      className="text-left"
    >
      <div className="card feature-card p-6">
        <div className={`text-3xl mb-4 ${card.color}`}>
          <i className={`fas fa-${card.icon}`}></i>
        </div>
        <h3 className="text-lg font-semibold text-gray-800
mb-2">{card.title}</h3>
        <p className="text-gray-600 text-sm mb-3">{card.desc}</p>
        <span className="text-xs font-medium bg-gray-100 text-gray-700 px-2
py-1 rounded-full">
          {card.stats}
        </span>
      </div>
    </button>
  )))
</div>

{/* Market Overview */}
<div className="bg-white rounded-2xl shadow-sm p-6">
  <div className="flex flex-wrap items-center justify-between mb-6">
    <h3 className="text-xl font-semibold text-gray-800 flex items-center">
      <i className="fas fa-chart-bar text-blue-500 mr-3"></i>
      Today's Market Prices
    </h3>
    <button
      onClick={() => setActiveTab('market')}
      className="text-green-600 text-sm hover:underline"
    >
      View All Market Data
    </button>
  </div>

  <div className="overflow-x-auto">

```

```

<table className="min-w-full divide-y divide-gray-200">
  <thead className="bg-gray-50">
    <tr>
      <th className="px-6 py-3 text-left text-xs font-medium text-gray-500 uppercase tracking-wider">Crop</th>
      <th className="px-6 py-3 text-left text-xs font-medium text-gray-500 uppercase tracking-wider">Quality</th>
      <th className="px-6 py-3 text-left text-xs font-medium text-gray-500 uppercase tracking-wider">Price</th>
      <th className="px-6 py-3 text-left text-xs font-medium text-gray-500 uppercase tracking-wider">Change</th>
      <th className="px-6 py-3 text-left text-xs font-medium text-gray-500 uppercase tracking-wider">Volume</th>
      <th className="px-6 py-3 text-left text-xs font-medium text-gray-500 uppercase tracking-wider">Forecast</th>
    </tr>
  </thead>
  <tbody className="bg-white divide-y divide-gray-200">
    {mockCrops.map((crop, index) => (
      <tr key={index} className="hover:bg-gray-50 transition">
        <td className="px-6 py-4 whitespace-nowrap">
          <div className="flex items-center">
            <div className="flex-shrink-0 h-10 w-10 bg-green-100 rounded-lg flex items-center justify-center">
              <i className="fas fa-seedling text-green-600"></i>
            </div>
            <div className="ml-4">
              <div className="text-sm font-medium text-gray-900">{crop.name}</div>
            </div>
          </div>
        </td>
        <td className="px-6 py-4 whitespace-nowrap">
          <span className="px-2 py-1 text-xs rounded-full bg-blue-100 text-blue-800">{crop.quality}</span>
        </td>
        <td className="px-6 py-4 whitespace-nowrap">
          <div className="text-sm font-semibold text-gray-900">{crop.price}</div>
          <div className="text-xs text-gray-500">{crop.unit}</div>
        </td>
        <td className="px-6 py-4 whitespace-nowrap">
          <span className={`flex items-center ${crop.trend === 'up' ? 'text-green-600' : 'text-red-600'} `}>
            <i className={`fas ${crop.trend === 'up' ? 'fa-arrow-up' : 'fa-arrow-down'} mr-1`}></i>
            {crop.change}
          </span>
        </td>
      </tr>
    ))}
  </tbody>
</table>

```

```

</td>
<td className="px-6 py-4 whitespace-nowrap text-sm
text-gray-500">
    {crop.volume}
</td>
<td className="px-6 py-4 whitespace-nowrap">
    <span className={`px-2 py-1 text-xs rounded-full font-medium
    ${
        crop.forecast === 'bullish' ? 'bg-green-100 text-green-800' :
        crop.forecast === 'stable' ? 'bg-blue-100 text-blue-800' :
        'bg-yellow-100 text-yellow-800'
    }`>
        {crop.forecast}
    </span>
</td>
</tr>
    )}}
</tbody>
</table>
</div>
</div>

```

```

{/* Weather & Insights */}
<div className="grid grid-cols-1 lg:grid-cols-2 gap-6">
    {/* Weather Card */}
    <div className="bg-white rounded-2xl shadow-sm p-6">
        <div className="flex justify-between items-center mb-4">
            <h3 className="text-xl font-semibold text-gray-800 flex items-center">
                <i className="fas fa-cloud-sun text-yellow-500 mr-3"></i>
                Weather Forecast
            </h3>
            <button
                onClick={() => {
                    setShowWeatherAlertForm(true);
                    setTimeout(() => {
                        weatherAlertRef.current?.scrollIntoView({ behavior: 'smooth' });
                    }, 100);
                }}
                className="btn-secondary px-3 py-1 rounded-lg text-sm"
            >
                <i className="fas fa-bell mr-1"></i>
                Set Alert
            </button>
        </div>

        <div className="grid grid-cols-3 gap-4 mb-4">
            <div className="bg-green-50 rounded-lg p-4 text-center">

```

```

        <div className="text-3xl font-bold text-gray-800
mb-1">{weatherData.temp}°C</div>
        <div className="text-sm text-gray-600">Temperature</div>
    </div>

    <div className="bg-blue-50 rounded-lg p-4 text-center">
        <div className="text-3xl font-bold text-gray-800
mb-1">{weatherData.humidity}%</div>
        <div className="text-sm text-gray-600">Humidity</div>
    </div>

    <div className="bg-purple-50 rounded-lg p-4 text-center">
        <div className="text-3xl font-bold text-gray-800
mb-1">{weatherData.windSpeed} km/h</div>
        <div className="text-sm text-gray-600">Wind Speed</div>
    </div>

    <div className="space-y-3 mb-4">
        {weatherData.forecast.map((day, index) => (
            <div key={index} className="flex items-center justify-between p-3
hover:bg-gray-50 rounded-lg">
                <div className="flex items-center space-x-3">
                    <span className="text-gray-700 font-medium
w-16">{day.day}</span>
                    <i className={`fas fa-${
                        day.condition === 'sunny' ? 'sun' :
                        day.condition === 'partly-cloudy' ? 'cloud-sun' :
                        day.condition === 'cloudy' ? 'cloud' : 'cloud-rain'
                    } text-yellow-500`}></i>
                </div>
                <div className="flex items-center space-x-4">
                    <span className="text-gray-600 text-sm">Rain:
{day.rain}%</span>
                    <div>
                        <span className="text-gray-800
font-medium">{day.high}°</span>
                        <span className="text-gray-500 mx-1"></span>
                        <span className="text-gray-500">{day.low}°</span>
                    </div>
                </div>
            </div>
        ))}
    </div>

    <div className="mt-4 pt-4 border-t border-gray-100">
        <p className="text-green-600 font-medium">

```

```

        <i className="fas fa-check-circle mr-2"></i>
        Good weather for field operations today
    </p>
</div>

{/* Weather Alerts */}
<div ref={weatherAlertRef}>
    {weatherAlerts.length > 0 && (
        <div className="mt-4 pt-4 border-t border-gray-100">
            <h4 className="text-sm font-medium text-gray-700 mb-2">Your
Weather Alerts</h4>
            <div className="space-y-2">
                {weatherAlerts.map(alert => (
                    <div key={alert.id} className="flex justify-between items-center
bg-blue-50 p-2 rounded">
                        <div>
                            <span className="text-sm font-medium">Alert when
{alert.condition} {alert.condition === 'rain' ? '>=' : '<='} {alert.value}{alert.unit}</span>
                        </div>
                        <div className="flex space-x-2">
                            <button
                                onClick={() => handleEditWeatherAlert(alert)}
                                className="text-blue-600 hover:text-blue-800 text-sm"
                            >
                                <i className="fas fa-edit"></i>
                            </button>
                            <button
                                onClick={() => handleRemoveWeatherAlert(alert.id)}
                                className="text-red-500 hover:text-red-700 text-sm"
                            >
                                <i className="fas fa-trash"></i>
                            </button>
                        </div>
                    </div>
                ))}
            </div>
        </div>
    )}
</div>

{/* Weather Alert Form */}
{showWeatherAlertForm && (
    <div className="mt-4 pt-4 border-t border-gray-100">
        <div className="flex justify-between items-center mb-2">
            <h4 className="text-sm font-medium text-gray-700">Set Weather
Alert</h4>
            <button
                onClick={() => setShowWeatherAlertForm(false)}
                className="text-gray-500 hover:text-gray-700"
            >

```

```

        >
        <i className="fas fa-times"></i>
      </button>
    </div>
    <div className="space-y-3">
      <div>
        <label className="block text-sm text-gray-600
mb-1">Condition</label>
        <select
          value={weatherAlertCondition}
          onChange={(e) => setWeatherAlertCondition(e.target.value)}
          className="w-full input-field py-2 px-3 rounded-lg"
        >
          <option value="rain">Rainfall</option>
          <option value="temperature">Temperature</option>
          <option value="humidity">Humidity</option>
        </select>
      </div>
      <div className="grid grid-cols-2 gap-3">
        <div>
          <label className="block text-sm text-gray-600
mb-1">Value</label>
          <input
            type="number"
            value={weatherAlertValue}
            onChange={(e) => setWeatherAlertValue(e.target.value)}
            className="w-full input-field py-2 px-3 rounded-lg"
          />
        </div>
        <div>
          <label className="block text-sm text-gray-600
mb-1">Unit</label>
          <select
            value={weatherAlertUnit}
            onChange={(e) => setWeatherAlertUnit(e.target.value)}
            className="w-full input-field py-2 px-3 rounded-lg"
          >
            {weatherAlertCondition === 'rain' ? (
              <option value="mm">mm</option>
            ) : weatherAlertCondition === 'temperature' ? (
              <>
                <option value="°C">°C</option>
                <option value="°F">°F</option>
              </>
            ) : (
              <option value="%">%</option>
            )}
          </select>

```

```

        </div>
      </div>
      <div className="flex justify-end space-x-2">
        <button
          onClick={() => setShowWeatherAlertForm(false)}
          className="bg-gray-100 text-gray-800 px-3 py-1 rounded-lg
hover:bg-gray-200"
        >
          Cancel
        </button>
        <button
          onClick={handleAddWeatherAlert}
          className="btn-primary px-3 py-1 rounded-lg"
        >
          Save Alert
        </button>
      </div>
    </div>
  </div>
)}
</div>
</div>

```

```

{ /* AI Insights */
<div className="bg-white rounded-2xl shadow-sm p-6">
  <h3 className="text-xl font-semibold text-gray-800 flex items-center mb-4">
    <i className="fas fa-lightbulb text-green-500 mr-3"></i>
    AI Farming Insights
  </h3>

  <div className="space-y-4">
    <div className="flex items-start">
      <div className="flex-shrink-0 mt-1">
        <div className="w-2 h-2 bg-green-500 rounded-full pulse"></div>
      </div>
      <div className="ml-3">
        <p className="text-gray-700">Optimal planting time for wheat in
next 7 days</p>
      </div>
    </div>

    <div className="flex items-start">
      <div className="flex-shrink-0 mt-1">
        <div className="w-2 h-2 bg-blue-500 rounded-full pulse"></div>
      </div>
      <div className="ml-3">
        <p className="text-gray-700">Rice prices expected to rise by 12%
this month</p>
      </div>
    </div>
  </div>

```



```

    </div>
  </div>

  <div className="flex items-start">
    <div className="flex-shrink-0 mt-1">
      <div className="w-2 h-2 bg-yellow-500 rounded-full pulse"></div>
    </div>
    <div className="ml-3">
      <p className="text-gray-700">Apply for PM-KISAN scheme -
deadline in 15 days</p>
    </div>
  </div>
</div>

<div className="mt-4 pt-4 border-t border-gray-100">
  <button
    onClick={() => setActiveTab('schemes')}
    className="btn-primary px-4 py-2 rounded-lg"
  >
    <i className="fas fa-file-invoice-dollar mr-2"></i>
    View Govt Schemes
  </button>
</div>
</div>
</div>

{/* Reminders List */}
{reminders.length > 0 && (
  <div className="bg-white rounded-2xl shadow-sm p-6">
    <h3 className="text-xl font-semibold text-gray-800 flex items-center mb-4">
      <i className="fas fa-bell text-yellow-500 mr-3"></i>
      Your Reminders
    </h3>

    <div className="space-y-3">
      {reminders.map((reminder) => (
        <div key={reminder.id} className="border border-gray-200 rounded-lg
p-4 flex justify-between items-center">
          <div>
            <p className="text-gray-800">{reminder.text}</p>
            <p className="text-gray-500 text-sm mt-1">{reminder.date}</p>
          </div>
          <button
            onClick={() => setReminders(reminders.filter(r => r.id !==
reminder.id))}
            className="text-red-500 hover:text-red-700"
          >
            <i className="fas fa-trash"></i>

```

```

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

```

// Enhanced Crop Disease Detection Component

```

const CropDiseaseDetection = ({ setActiveTab }) => {
  const [selectedImage, setSelectedImage] = useState(null);
  const [isAnalyzing, setIsAnalyzing] = useState(false);
  const [results, setResults] = useState(null);
  const [confidence, setConfidence] = useState(0);
  const fileInputRef = useRef(null);

```

```

  const handleImageUpload = (event) => {
    const file = event.target.files[0];
    if (file) {
      const reader = new FileReader();
      reader.onload = (e) => {
        setSelectedImage(e.target.result);
      };
      reader.readAsDataURL(file);
    }
  };

```

```

  const analyzeImage = () => {
    setIsAnalyzing(true);
    setResults(null);

```

// Simulate AI analysis

```

  setTimeout(() => {
    const mockResults = [
      {
        disease: "Leaf Blight",
        probability: 87,
        severity: "Moderate",
        treatment: "Apply copper-based fungicide within 24 hours",
        prevention: "Ensure proper drainage and avoid overhead watering"
      },
      {
        disease: "Healthy Plant",
        probability: 13,
        severity: "None",
        treatment: "Continue regular care routine",

```

```

        prevention: "Maintain current practices"
      }
    ];
    setResults(mockResults);
    setConfidence(87);
    setIsAnalyzing(false);
  }, 3000);
};

return (
  <div className="container mx-auto px-4 py-8 space-y-8 slide-in">
    <div className="back-button" onClick={() => setActiveTab('dashboard')}>
      <i className="fas fa-arrow-left"></i>
      <span>Back to Dashboard</span>
    </div>

    <div className="text-center">
      <h2 className="text-2xl md:text-3xl font-bold text-gray-800 mb-2">
        AI Crop Disease Detection
      </h2>
      <p className="text-gray-600 max-w-2xl mx-auto">
        Upload a photo of your crop and get instant diagnosis with treatment
        recommendations
      </p>
    </div>

    <div className="grid grid-cols-1 lg:grid-cols-2 gap-8">
      { /* Image Upload Section */ }
      <div className="bg-white rounded-2xl shadow-sm p-6">
        <h3 className="text-lg font-semibold text-gray-800 mb-4 flex items-center">
          <i className="fas fa-camera text-green-500 mr-3"></i>
          Upload Crop Image
        </h3>

        <div
          className="border-2 border-dashed border-gray-300 rounded-xl p-8
            text-center cursor-pointer transition hover:border-green-500 hover:bg-green-50"
          onClick={() => fileInputRef.current?.click()}
        >
          {selectedImage ? (
            <div className="space-y-4">
              <img
                src={selectedImage}
                alt="Uploaded crop"
                className="w-full h-64 object-contain rounded-lg"
              />
              <p className="text-green-600 font-medium">Image uploaded
                successfully</p>
            </div>
          ) : (
            <p>Upload a photo of your crop to get an instant diagnosis</p>
          )}
        </div>
      </div>
    </div>
  </div>
);

```

```

        </div>
      ) : (
        <div className="space-y-4">
          <div className="w-16 h-16 bg-green-100 rounded-2xl flex
items-center justify-center mx-auto">
            <i className="fas fa-plus text-green-500 text-2xl"></i>
          </div>
          <div>
            <p className="text-gray-800 font-medium mb-2">Click to upload
crop image</p>
            <p className="text-gray-500 text-sm">Supports JPG, PNG up to
10MB</p>
          </div>
        </div>
      )}
    </div>

    <input
      ref={fileInputRef}
      type="file"
      accept="image/*"
      onChange={handleImageUpload}
      className="hidden"
    />

    {selectedImage && (
      <div className="flex space-x-3 mt-4">
        <button
          onClick={() => fileInputRef.current?.click()}
          className="flex-1 bg-gray-100 text-gray-800 py-3 px-6 rounded-lg
font-medium hover:bg-gray-200 transition"
        >
          <i className="fas fa-sync-alt mr-2"></i>
          Change Image
        </button>
        <button
          onClick={analyzeImage}
          disabled={isAnalyzing}
          className="flex-1 btn-primary py-3 px-6 rounded-lg font-medium
disabled:opacity-70"
        >
          {isAnalyzing ? (
            <>
              <i className="fas fa-spinner fa-spin mr-2"></i>
              Analyzing...
            </>
          ) : (
            <>

```

```

        <i className="fas fa-search mr-2"></i>
        Analyze
    </>
    })
</button>
</div>
})
</div>

{/* Results Section */}
<div className="bg-white rounded-2xl shadow-sm p-6">
    <h3 className="text-lg font-semibold text-gray-800 mb-4 flex items-center">
        <i className="fas fa-chart-pie text-blue-500 mr-3"></i>
        Analysis Results
    </h3>

    {isAnalyzing && (
        <div className="space-y-4">
            <div className="flex items-center space-x-3">
                <div className="w-8 h-8 border-2 border-blue-500
border-t-transparent rounded-full animate-spin"></div>
                <span className="text-gray-600">AI is analyzing your crop
image...</span>
            </div>
            <div className="space-y-3">
                {[1, 2, 3].map((i) => (
                    <div key={i} className="h-4 bg-gray-200 rounded-lg
animate-pulse"></div>
                ))}
            </div>
        </div>
    )}

    {results && !isAnalyzing && (
        <div className="space-y-6">
            <div className="bg-blue-50 rounded-xl p-4">
                <div className="flex items-center justify-between mb-3">
                    <span className="text-gray-700 font-medium">Confidence
Level</span>
                    <span className="text-green-600
font-bold">{confidence}%</span>
                </div>
                <div className="w-full bg-gray-200 rounded-full h-2">
                    <div
                        className="bg-green-500 h-2 rounded-full"
                        style={{ width: `${confidence}%` }}
                    ></div>
                </div>
            </div>
        </div>
    )}

```

```

</div>

<div className="space-y-4">
  {results.map((result, index) => (
    <div key={index} className="border border-gray-200 rounded-xl
p-4">
      <div className="flex items-start justify-between mb-3">
        <div>
          <h4 className="text-gray-800
font-semibold">{result.disease}</h4>
          <p className="text-gray-500 text-sm">Probability:
{result.probability}%</p>
        </div>
        <div className={`px-3 py-1 rounded-full text-xs font-medium
${
      result.severity === 'None' ? 'bg-green-100 text-green-800' :
      result.severity === 'Moderate' ? 'bg-yellow-100
text-yellow-800' :
      'bg-red-100 text-red-800'
    }`}>
          {result.severity}
        </div>
      </div>
      <div className="space-y-2">
        <div>
          <span className="text-gray-700 text-sm
font-medium">Treatment: </span>
          <span className="text-gray-600
text-sm">{result.treatment}</span>
        </div>
        <div>
          <span className="text-gray-700 text-sm
font-medium">Prevention: </span>
          <span className="text-gray-600
text-sm">{result.prevention}</span>
        </div>
      </div>
    </div>
  )})
</div>
</div>

{!selectedImage && !isAnalyzing && (
  <div className="text-center py-12">
    <div className="w-16 h-16 bg-gray-100 rounded-2xl flex items-center
justify-center mx-auto mb-4">
      <i className="fas fa-microscope text-gray-400 text-2xl"></i>
    </div>
  </div>
)}

```

```

        </div>
        <p className="text-gray-500">Upload an image to see AI analysis
results</p>
    </div>
    })
  </div>
</div>
</div>
);
};

```

// Market Intelligence Component

```

const MarketIntelligence = ({ setActiveTab }) => {
  const [selectedMarket, setSelectedMarket] = useState('all');
  const [priceAlerts, setPriceAlerts] = useState([]);
  const [showAlertForm, setShowAlertForm] = useState(false);
  const [alertCrop, setAlertCrop] = useState("");
  const [alertPrice, setAlertPrice] = useState("");
  const [alertCondition, setAlertCondition] = useState('above');
  const [editingAlertId, setEditingAlertId] = useState(null);
  const alertSectionRef = useRef(null);

  const markets = [
    { id: 'all', name: 'All Markets', count: 500 },
    { id: 'ap', name: 'Andhra Pradesh', count: 45 },
    { id: 'tn', name: 'Tamil Nadu', count: 38 },
    { id: 'ka', name: 'Karnataka', count: 42 },
    { id: 'mh', name: 'Maharashtra', count: 55 }
  ];

  const handleSetAlert = (crop) => {
    setAlertCrop(crop.name);
    setAlertPrice(crop.price.replace("$", "").replace(", ", ""));
    setShowAlertForm(true);
    setEditingAlertId(null);
    // Scroll to alert section
    setTimeout(() => {
      alertSectionRef.current?.scrollIntoView({ behavior: 'smooth' });
    }, 100);
  };

  const handleEditAlert = (alert) => {
    setAlertCrop(alert.crop);
    setAlertPrice(alert.price);
    setAlertCondition(alert.condition);
    setShowAlertForm(true);
    setEditingAlertId(alert.id);
    setTimeout(() => {

```

```

        alertSectionRef.current?.scrollIntoView({ behavior: 'smooth' });
    }, 100);
};

const handleSaveAlert = () => {
    if (alertCrop && alertPrice) {
        if (editingAlertId) {
            // Update existing alert
            setPriceAlerts(priceAlerts.map(alert =>
                alert.id === editingAlertId ?
                { ...alert, crop: alertCrop, price: alertPrice, condition: alertCondition } :
                alert
            ));
        } else {
            // Add new alert
            setPriceAlerts([...priceAlerts, {
                id: Date.now(),
                crop: alertCrop,
                price: alertPrice,
                condition: alertCondition
            }]);
        }
        setShowAlertForm(false);
        setAlertCrop("");
        setAlertPrice("");
        setEditingAlertId(null);
    }
};

const handleRemoveAlert = (id) => {
    setPriceAlerts(priceAlerts.filter(alert => alert.id !== id));
};

return (
    <div className="container mx-auto px-4 py-8 space-y-8 slide-in">
        <div className="back-button" onClick={() => setActiveTab('dashboard')}>
            <i className="fas fa-arrow-left"></i>
            <span>Back to Dashboard</span>
        </div>

        <div className="text-center">
            <h2 className="text-2xl md:text-3xl font-bold text-gray-800 mb-2">
                Market Intelligence
            </h2>
            <p className="text-gray-600 max-w-2xl mx-auto">
                Real-time market data from 500+ mandis across India with price predictions
            </p>
        </div>
    </div>

```



```

    { /* Market Filters */
    <div className="bg-white rounded-2xl shadow-sm p-6">
      <div className="flex flex-wrap gap-2">
        {markets.map((market) => (
          <button
            key={market.id}
            onClick={() => setSelectedMarket(market.id)}
            className={`tab-button ${selectedMarket === market.id ? 'active' : ''}`}
          >
            {market.name}
            <span className="ml-1 text-xs opacity-70">({market.count})</span>
          </button>
        ))}
      </div>
    </div>

    { /* Price Cards */
    <div className="grid grid-cols-1 md:grid-cols-2 gap-6">
      {mockCrops.map((crop, index) => (
        <div key={index} className="card bg-white rounded-2xl shadow-sm p-6">
          <div className="flex items-center justify-between mb-4">
            <div className="flex items-center space-x-4">
              <div className="w-12 h-12 bg-green-100 rounded-xl flex
items-center justify-center">
                <i className="fas fa-seedling text-green-600"></i>
              </div>
            </div>
            <div>
              <h3 className="text-gray-800 font-semibold">{crop.name}</h3>
              <p className="text-gray-500 text-sm">{crop.quality}</p>
            </div>
          </div>
          <div className="text-right">
            <div className="text-xl font-bold text-gray-800">{crop.price}</div>
            <div className="text-gray-500 text-sm">{crop.unit}</div>
          </div>
        </div>

        <div className="space-y-3">
          <div className="flex items-center justify-between">
            <span className="text-gray-600 text-sm">24h Change</span>
            <span className={`font-semibold ${crop.trend === 'up' ?
'text-green-600' : 'text-red-600'}`}>>
              {crop.change}
            </span>
          </div>
          <div className="flex items-center justify-between">
            <span className="text-gray-600 text-sm">Volume</span>

```

```

        <span className="text-gray-800
font-medium">{crop.volume}</span>
      </div>
      <div className="flex items-center justify-between">
        <span className="text-gray-600 text-sm">Forecast</span>
        <span className={`px-2 py-1 rounded-full text-xs font-medium ${
          crop.forecast === 'bullish' ? 'bg-green-100 text-green-800' :
          crop.forecast === 'stable' ? 'bg-blue-100 text-blue-800' :
          'bg-yellow-100 text-yellow-800'
        }`}>
          {crop.forecast}
        </span>
      </div>
    </div>

    <div className="mt-4 pt-4 border-t border-gray-100">
      <button
        onClick={() => handleSetAlert(crop)}
        className="btn-secondary w-full py-2 px-4 rounded-lg"
      >
        <i className="fas fa-bell mr-2"></i>
        Set Price Alert
      </button>
    </div>
  </div>
  )})
</div>

```

```

{/* Price Alerts Section */}
<div className="bg-white rounded-2xl shadow-sm p-6" ref={alertSectionRef}>
  <div className="flex justify-between items-center mb-4">
    <h3 className="text-xl font-semibold text-gray-800 flex items-center">
      <i className="fas fa-bell text-yellow-500 mr-3"></i>
      Your Price Alerts
    </h3>
    <button
      onClick={() => {
        setShowAlertForm(true);
        setAlertCrop("");
        setAlertPrice("");
        setEditingAlertId(null);
        setTimeout(() => {
          alertSectionRef.current?.scrollIntoView({ behavior: 'smooth' });
        }, 100);
      }}
      className="btn-primary px-4 py-2 rounded-lg"
    >
      <i className="fas fa-plus mr-2"></i>

```

```

        Add New Alert
      </button>
    </div>

    {priceAlerts.length > 0 ? (
      <div className="alert-section space-y-3">
        {priceAlerts.map((alert) => (
          <div key={alert.id} className="border border-gray-200 rounded-lg
p-4">
          <div className="flex justify-between items-center mb-2">
            <p className="text-gray-800 font-medium">{alert.crop}</p>
            <div className="flex space-x-2">
              <button
                onClick={() => handleEditAlert(alert)}
                className="text-blue-600 hover:text-blue-800"
              > <i className="fas fa-edit"></i>
            </button>
            <button
              onClick={() => handleRemoveAlert(alert.id)}
              className="text-red-500 hover:text-red-700"
            >
              <i className="fas fa-trash"></i>
            </button>
          </div>
        </div>
        <div>
          <p className="text-sm text-gray-600">
            Alert when price goes {alert.condition} ${alert.price}
          </p>
        </div>
      </div>
    ) : (
      <div className="empty-state">
        <div className="empty-state-icon">
          <i className="fas fa-bell-slash text-gray-400"></i>
        </div>
        <h4 className="text-gray-700 font-medium mb-2">No price alerts set</h4>
        <p className="text-gray-500 max-w-md mx-auto mb-4">
          Set alerts to get notified when crop prices reach your desired level
        </p>
        <button
          onClick={() => setShowAlertForm(true)}
          className="btn-primary px-4 py-2 rounded-lg"
        >
          <i className="fas fa-plus mr-2"></i>
          Create Alert
        </button>
      </div>
    ) : (

```

```

    })

    { /* Price Alert Form */
    { showAlertForm && (
      <div className="mt-6 pt-6 border-t border-gray-200">
        <div className="flex justify-between items-center mb-4">
          <h4 className="text-lg font-semibold text-gray-800">
            {editingAlertId ? 'Edit Price Alert' : 'Create New Price Alert'}
          </h4>
          <button
            onClick={() => {
              setShowAlertForm(false);
              setEditingAlertId(null);
            }}
            className="text-gray-500 hover:text-gray-700"
          >
            <i className="fas fa-times"></i>
          </button>
        </div>

        <div className="space-y-4">
          <div>
            <label className="block text-sm font-medium text-gray-700
mb-1">Crop</label>
            <input
              type="text"
              value={alertCrop}
              onChange={(e) => setAlertCrop(e.target.value)}
              className="w-full input-field py-2 px-3 rounded-lg"
              placeholder="Enter crop name"
            />
          </div>

          <div className="grid grid-cols-1 md:grid-cols-2 gap-4">
            <div>
              <label className="block text-sm font-medium text-gray-700
mb-1">Price</label>
              <div className="relative">
                <span className="absolute left-3 top-1/2 transform
-rotate-y-1/2 text-gray-500">$</span>
                <input
                  type="number"
                  value={alertPrice}
                  onChange={(e) => setAlertPrice(e.target.value)}
                  className="w-full input-field py-2 px-8 rounded-lg"
                  placeholder="Enter price"
                />
              </div>
            </div>

```



```
const [selectedLanguage, setSelectedLanguage] = useState('hi-IN');
const messagesEndRef = useRef(null);
```

```
const languages = [
  { code: 'hi-IN', name: 'Hindi', flag: '🇮🇳' },
  { code: 'en-IN', name: 'English', flag: '🇮🇳' },
  { code: 'ta-IN', name: 'Tamil', flag: '🇮🇳' },
  { code: 'te-IN', name: 'Telugu', flag: '🇮🇳' },
  { code: 'mr-IN', name: 'Marathi', flag: '🇮🇳' },
  { code: 'bn-IN', name: 'Bengali', flag: '🇮🇳' }
];
```

```
useEffect(() => {
  if (messagesEndRef.current) {
    messagesEndRef.current.scrollToView({ behavior: 'smooth' });
  }
}, [messages]);
```

```
const handleSendMessage = () => {
  if (transcript.trim()) {
    const userMessage = {
      id: Date.now(),
      text: transcript,
      sender: 'user',
      timestamp: new Date().toLocaleTimeString()
    };
  }
```

```
  setMessages([...messages, userMessage]);
  setIsProcessing(true);
```

```
  // Simulate AI response
```

```
  setTimeout(() => {
```

```
    const aiResponses = [
```

```
      "मैं आपकी फसल की समस्या समझ गया हूँ। यह लीफ ब्लाइट रोग लगता है। तुरंत कॉपर आधारित फंगीसाइड का छिड़काव करें।",
```

```
      "आपके क्षेत्र में गेहूं की कीमतें अगले 15 दिनों में 8-10% बढ़ने की संभावना है।",
```

```
      "आपके मिट्टी के नमूने के अनुसार, अगली फसल के लिए NPK 10:26:26 उर्वरक की सिफारिश की जाती है।",
```

```
      "आपके क्षेत्र में कल भारी बारिश की चेतावनी है। कृपया अपनी फसल को ढक कर रखें।"
```

```
    ];
```

```
    const randomResponse = aiResponses[Math.floor(Math.random() * aiResponses.length)];
```

```
    const aiMessage = {
```

```
      id: Date.now() + 1,
```

```
      text: randomResponse,
```

```
      sender: 'ai',
```

```

        timestamp: new Date().toLocaleTimeString()
    };

    setMessages(prev => [...prev, aiMessage]);
    setIsProcessing(false);
  }, 2000);
}
};

const toggleListening = () => {
  if (isListening) {
    stopListening();
    if (transcript.trim()) {
      handleSendMessage();
    }
  } else {
    startListening();
  }
};

return (
  <div className="container mx-auto px-4 py-8 space-y-8 slide-in">
    <div className="back-button" onClick={() => setActiveTab('dashboard')}>
      <i className="fas fa-arrow-left"></i>
      <span>Back to Dashboard</span>
    </div>

    <div className="text-center">
      <h2 className="text-2xl md:text-3xl font-bold text-gray-800 mb-2">
        Voice Assistant
      </h2>
      <p className="text-gray-600 max-w-2xl mx-auto">
        Ask questions in your local language and get instant farming advice
      </p>
    </div>

    <div className="bg-white rounded-2xl shadow-sm p-6">
      { /* Language Selector */ }
      <div className="mb-6">
        <label className="block text-sm font-medium text-gray-700 mb-2">Select
Language</label>
        <div className="flex flex-wrap gap-2">
          {languages.map((lang) => (
            <button
              key={lang.code}
              onClick={() => setSelectedLanguage(lang.code)}
              className={`tab-button ${selectedLanguage === lang.code ? 'active' :

```

```

        >
        {lang.flag} {lang.name}
      </button>
    )))
  </div>
</div>

{/* Chat Container */}
<div className="bg-gray-50 rounded-xl p-4 h-96 overflow-y-auto mb-4">
  {messages.length === 0 ? (
    <div className="h-full flex flex-col items-center justify-center text-center">
      <div className="w-16 h-16 bg-green-100 rounded-2xl flex items-center
justify-center mb-4">
        <i className="fas fa-microphone text-green-600 text-2xl"></i>
      </div>
      <h4 className="text-gray-700 font-medium mb-2">Ask your farming
question</h4>
      <p className="text-gray-500 max-w-md">
        Press the microphone button and speak in your language to get instant
farming advice
      </p>
    </div>
  ) : (
    <div className="space-y-4">
      {messages.map((message) => (
        <div
          key={message.id}
          className={`flex ${message.sender === 'user' ? 'justify-end' :
'justify-start'}`}
        >
          <div
            className={`max-w-xs md:max-w-md rounded-lg p-3 ${
              message.sender === 'user'
                ? 'bg-green-500 text-white rounded-br-none'
                : 'bg-gray-200 text-gray-800 rounded-bl-none'
            }`}
          >
            <p>{message.text}</p>
            <p className="text-xs opacity-70
mt-1">{message.timestamp}</p>
          </div>
        </div>
      ))}
      {isProcessing && (
        <div className="flex justify-start">
          <div className="bg-gray-200 text-gray-800 rounded-lg
rounded-bl-none p-3 max-w-xs">
            <div className="flex space-x-2">

```



```

        <div className="w-2 h-2 bg-gray-500 rounded-full
animate-bounce"></div>
        <div className="w-2 h-2 bg-gray-500 rounded-full
animate-bounce" style={{ animationDelay: '0.2s' }}></div>
        <div className="w-2 h-2 bg-gray-500 rounded-full
animate-bounce" style={{ animationDelay: '0.4s' }}></div>
    </div>
</div>
</div>
    )}
    <div ref={messagesEndRef} />
</div>
    )}
</div>

{/* Voice Input */}
<div className="flex items-center space-x-3">
  <div className="flex-1 bg-white border border-gray-300 rounded-lg p-3">
    {isListening ? (
      <div className="flex items-center space-x-2">
        <div className="flex space-x-1">
          {[1, 2, 3, 4, 5].map((i) => (
            <div
              key={i}
              className="w-1 bg-green-500 rounded-full voice-wave"
              style={{
                animationDelay: `${i * 0.1}s`,
                height: `${8 + Math.random() * 16}px`
              }}
            ></div>
          ))}
        </div>
        <span className="text-gray-600">{transcript || 'Listening...'}</span>
      </div>
    ) : (
      <span className="text-gray-400">Press microphone to speak</span>
    )}
  </div>
  <button
    onClick={toggleListening}
    className={`w-12 h-12 rounded-full flex items-center justify-center text-xl ${
      isListening
        ? 'bg-red-500 text-white animate-pulse'
        : 'bg-green-500 text-white'
    }`}
  >
    <i className={`fas ${isListening ? 'fa-stop' : 'fa-microphone'}`}></i>
  </button>

```

```

</div>

{/* Confidence Meter */}
{isListening && (
  <div className="mt-4">
    <div className="flex justify-between text-xs text-gray-500 mb-1">
      <span>Confidence: {Math.round(confidence * 100)}%</span>
      <span>{selectedLanguage}</span>
    </div>
    <div className="w-full bg-gray-200 rounded-full h-2">
      <div
        className="bg-green-500 h-2 rounded-full"
        style={{ width: `${confidence * 100}%` }}
      ></div>
    </div>
  </div>
)}
</div>

{/* Sample Questions */}
<div className="bg-white rounded-2xl shadow-sm p-6">
  <h3 className="text-lg font-semibold text-gray-800 mb-4 flex items-center">
    <i className="fas fa-lightbulb text-yellow-500 mr-3"></i>
    Try asking...
  </h3>
  <div className="grid grid-cols-1 md:grid-cols-2 gap-3">
    {[
      "मेरी फसल में पीले धब्बे क्यों हैं?",
      "गेहूं की कीमत कितनी है?",
      "मेरी मिट्टी के लिए कौन सा उर्वरक सबसे अच्छा है?",
      "आज मौसम कैसा रहेगा?"
    ].map((question, index) => (
      <button
        key={index}
        onClick={() => {
          setMessages([]);
          setTranscript(question);
        }}
        className="text-left bg-gray-50 hover:bg-gray-100 rounded-lg p-3 text-sm
text-gray-700 transition"
      >
        {question}
      </button>
    ))}
  </div>
</div>
</div>
);

```

```
};
```

```
// Crop Recommendations Component
```

```
const CropRecommendations = ({ setActiveTab }) => {  
  const [activeCrop, setActiveCrop] = useState('wheat');  
  const [soilType, setSoilType] = useState('loamy');  
  const [irrigation, setIrrigation] = useState('medium');  
  const [region, setRegion] = useState('north');  
  const [showFullSchedule, setShowFullSchedule] = useState(false);
```

```
  
  const crops = [  
    { id: 'wheat', name: 'Wheat', icon: 'wheat-awn' },  
    { id: 'rice', name: 'Rice', icon: 'rice' },  
    { id: 'corn', name: 'Corn', icon: 'corn' },  
    { id: 'cotton', name: 'Cotton', icon: 'cotton' },  
    { id: 'sugarcane', name: 'Sugarcane', icon: 'sugarcane' }  
  ];
```

```
  
  const soilTypes = [  
    { id: 'sandy', name: 'Sandy' },  
    { id: 'loamy', name: 'Loamy' },  
    { id: 'clay', name: 'Clay' },  
    { id: 'black', name: 'Black' },  
    { id: 'red', name: 'Red' }  
  ];
```

```
  
  const irrigationOptions = [  
    { id: 'low', name: 'Low' },  
    { id: 'medium', name: 'Medium' },  
    { id: 'high', name: 'High' }  
  ];
```

```
  
  const regions = [  
    { id: 'north', name: 'North' },  
    { id: 'south', name: 'South' },  
    { id: 'east', name: 'East' },  
    { id: 'west', name: 'West' }  
  ];
```

```
  
  const getCropSchedule = () => {  
    // This would normally come from an API based on the selected parameters  
    const baseSchedule = [  
      { day: 0, task: 'Land preparation and soil treatment', progress: 100 },  
      { day: 7, task: 'Sowing seeds with recommended spacing', progress: 100 },  
      { day: 14, task: 'First irrigation and weed control', progress: 100 },  
      { day: 21, task: 'First fertilizer application', progress: 100 },  
      { day: 30, task: 'Second irrigation and pest monitoring', progress: 80 },  
      { day: 45, task: 'Second weeding and fertilizer application', progress: 60 },
```

```

    { day: 60, task: 'Disease monitoring and third irrigation', progress: 40 },
    { day: 90, task: 'Final irrigation and pre-harvest preparations', progress: 20 },
    { day: 120, task: 'Harvesting and post-harvest management', progress: 0 }
  ];

  return showFullSchedule ? baseSchedule : baseSchedule.slice(0, 5);
};

return (
  <div className="container mx-auto px-4 py-8 space-y-8 slide-in">
    <div className="back-button" onClick={() => setActiveTab('dashboard')}>
      <i className="fas fa-arrow-left"></i>
      <span>Back to Dashboard</span>
    </div>

    <div className="text-center">
      <h2 className="text-2xl md:text-3xl font-bold text-gray-800 mb-2">
        Crop Advisor
      </h2>
      <p className="text-gray-600 max-w-2xl mx-auto">
        Get personalized crop recommendations and cultivation schedule based on
        your farm conditions
      </p>
    </div>

    <div className="grid grid-cols-1 lg:grid-cols-3 gap-8">
      { /* Left Column - Filters */ }
      <div className="bg-white rounded-2xl shadow-sm p-6">
        <h3 className="text-lg font-semibold text-gray-800 mb-4 flex items-center">
          <i className="fas fa-sliders-h text-green-500 mr-3"></i>
          Farm Parameters
        </h3>

        <div className="space-y-6">
          { /* Crop Selection */ }
          <div>
            <label className="block text-sm font-medium text-gray-700 mb-2">Select
            Crop</label>

            <div className="grid grid-cols-2 gap-2">
              {crops.map((crop) => (
                <button
                  key={crop.id}
                  onClick={() => setActiveCrop(crop.id)}
                  className={`flex items-center justify-center space-x-2 py-2 px-3
                    rounded-lg ${
                      activeCrop === crop.id
                        ? 'bg-green-500 text-white'
                        : 'bg-gray-100 text-gray-700 hover:bg-gray-200'
                    }`}
                >

```

```

        }}
      >
        <i className={`fas fa-${crop.icon}`}></i>
        <span>{crop.name}</span>
      </button>
    )}}
  </div>
</div>

{/* Soil Type */}
<div>
  <label className="block text-sm font-medium text-gray-700 mb-2">Soil
Type</label>
  <div className="flex flex-wrap gap-2">
    {soilTypes.map((soil) => (
      <button
        key={soil.id}
        onClick={() => setSoilType(soil.id)}
        className={`py-1 px-3 rounded-full text-sm ${
          soilType === soil.id
            ? 'bg-green-500 text-white'
            : 'bg-gray-100 text-gray-700 hover:bg-gray-200'
        }}
      >
        {soil.name}
      </button>
    ))}
  </div>
</div>

{/* Irrigation */}
<div>
  <label className="block text-sm font-medium text-gray-700
mb-2">Irrigation Availability</label>
  <div className="flex flex-wrap gap-2">
    {irrigationOptions.map((option) => (
      <button
        key={option.id}
        onClick={() => setIrrigation(option.id)}
        className={`py-1 px-3 rounded-full text-sm ${
          irrigation === option.id
            ? 'bg-green-500 text-white'
            : 'bg-gray-100 text-gray-700 hover:bg-gray-200'
        }}
      >
        {option.name}
      </button>
    ))}
  </div>
</div>

```

```

        </div>
    </div>

    { /* Region */}
    <div>
        <label className="block text-sm font-medium text-gray-700
mb-2">Region</label>
        <div className="flex flex-wrap gap-2">
            {regions.map((reg) => (
                <button
                    key={reg.id}
                    onClick={() => setRegion(reg.id)}
                    className={`py-1 px-3 rounded-full text-sm ${
                        region === reg.id
                            ? 'bg-green-500 text-white'
                            : 'bg-gray-100 text-gray-700 hover:bg-gray-200'
                    }`}
                >
                    {reg.name}
                </button>
            ))}
        </div>
    </div>
</div>
</div>

{ /* Middle Column - Recommendations */}
<div className="bg-white rounded-2xl shadow-sm p-6">
    <h3 className="text-lg font-semibold text-gray-800 mb-4 flex items-center">
        <i className="fas fa-seedling text-green-500 mr-3"></i>
        Recommendations
    </h3>

    <div className="space-y-6">
        <div className="bg-green-50 rounded-xl p-4">
            <h4 className="text-green-800 font-medium mb-2">Best Variety</h4>
            <p>PBW 343 (For {activeCrop === 'wheat' ? 'Wheat' : activeCrop === 'rice'
? 'Rice' : 'Corn'})</p>
        </div>

        <div>
            <h4 className="text-gray-700 font-medium mb-2">Fertilizer Plan</h4>
            <div className="space-y-3">
                <div className="flex justify-between items-center">
                    <span className="text-gray-600">Nitrogen (N)</span>
                    <span className="font-medium">120 kg/ha</span>
                </div>
                <div className="flex justify-between items-center">

```

```

        <span className="text-gray-600">Phosphorus (P)</span>
        <span className="font-medium">60 kg/ha</span>
      </div>
      <div className="flex justify-between items-center">
        <span className="text-gray-600">Potassium (K)</span>
        <span className="font-medium">40 kg/ha</span>
      </div>
    </div>
  </div>

  <div>
    <h4 className="text-gray-700 font-medium mb-2">Pesticide
Advisory</h4>
    <div className="space-y-2">
      <p className="text-sm">Pre-sowing: Chlorpyrifos 20 EC @ 2.5
l/ha</p>
      <p className="text-sm">Vegetative stage: Imidacloprid 17.8 SL @ 0.3
l/ha</p>
    </div>
  </div>

  <div>
    <h4 className="text-gray-700 font-medium mb-2">Expected Yield</h4>
    <div className="bg-blue-50 rounded-xl p-4 text-center">
      <div className="text-2xl font-bold text-blue-800">5.2 - 6.8
tons/ha</div>
      <div className="text-sm text-blue-600">Based on your
parameters</div>
    </div>
  </div>
</div>

{/* Right Column - Schedule */}
<div className="bg-white rounded-2xl shadow-sm p-6">
  <div className="flex justify-between items-center mb-4">
    <h3 className="text-lg font-semibold text-gray-800 flex items-center">
      <i className="fas fa-calendar-alt text-green-500 mr-3"></i>
      Cultivation Schedule
    </h3>
    <button
      onClick={() => setShowFullSchedule(!showFullSchedule)}
      className="text-green-600 text-sm hover:underline"
    >
      {showFullSchedule ? 'Show Less' : 'Show Full Schedule'}
    </button>
  </div>
</div>

```

```

    <div className="timeline">
      {getCropSchedule().map((stage, index) => (
        <div key={index} className="timeline-item">
          <div className="timeline-date">Day {stage.day}</div>
          <div className="timeline-content">
            <div className="flex justify-between items-center mb-1">
              <span className="font-medium">{stage.task}</span>
              <span className="text-sm
text-gray-500">{stage.progress}%</span>
            </div>
            <div className="w-full bg-gray-200 rounded-full h-2">
              <div
                className="bg-green-500 h-2 rounded-full"
                style={{ width: `${stage.progress}%` }}
              ></div>
            </div>
          </div>
        </div>
      ))}
    </div>
  </div>
</div>
</div>
);
};

```

// Government Schemes Component

```

const GovernmentSchemes = ({ setActiveTab }) => {
  const [selectedScheme, setSelectedScheme] = useState(null);
  const [searchQuery, setSearchQuery] = useState("");

  const filteredSchemes = mockSchemes.filter(scheme =>
    scheme.name.toLowerCase().includes(searchQuery.toLowerCase()) ||
    scheme.description.toLowerCase().includes(searchQuery.toLowerCase())
  );

  if (selectedScheme) {
    return (
      <div className="full-page-view">
        <div className="back-button" onClick={() => setSelectedScheme(null)}>
          <i className="fas fa-arrow-left"></i>
          <span>Back to Schemes</span>
        </div>

        <div className="bg-white rounded-2xl shadow-sm p-6">
          <div className="flex items-start justify-between mb-6">
            <div>

```



```

                <h2 className="text-2xl font-bold
text-gray-800">{selectedScheme.name}</h2>
                <div className="flex items-center space-x-4 mt-2">
                    <span className="bg-green-100 text-green-800 px-2 py-1 rounded-full
text-xs">
                        {selectedScheme.type}
                    </span>
                    <span className="bg-blue-100 text-blue-800 px-2 py-1 rounded-full
text-xs">
                        {selectedScheme.status}
                    </span>
                    <span className="text-gray-500 text-sm">
                        Beneficiaries: {selectedScheme.beneficiaries}
                    </span>
                </div>
            </div>
            <div className="w-16 h-16 bg-green-100 rounded-xl flex items-center
justify-center">
                <i className={`fas fa-${selectedScheme.icon} text-green-600
text-2xl`} ></i>
            </div>
        </div>

        <div className="space-y-6">
            <div>
                <h3 className="text-lg font-semibold text-gray-800
mb-2">Description</h3>
                <p className="text-gray-600">{selectedScheme.description}</p>
            </div>

            <div>
                <h3 className="text-lg font-semibold text-gray-800 mb-2">Eligibility</h3>
                <p className="text-gray-600">{selectedScheme.eligibility}</p>
            </div>

            <div>
                <h3 className="text-lg font-semibold text-gray-800 mb-2">Required
Documents</h3>
                <p className="text-gray-600">{selectedScheme.documents}</p>
            </div>

            <div className="pt-4 border-t border-gray-200">
                <button className="btn-primary w-full py-3 px-4 rounded-lg">
                    <i className="fas fa-file-signature mr-2"></i>
                    Apply Now
                </button>
            </div>
        </div>
    </div>

```

```

        </div>
      </div>
    );
  }

  return (
    <div className="container mx-auto px-4 py-8 space-y-8 slide-in">
      <div className="back-button" onClick={() => setActiveTab('dashboard')}>
        <i className="fas fa-arrow-left"></i>
        <span>Back to Dashboard</span>
      </div>

      <div className="text-center">
        <h2 className="text-2xl md:text-3xl font-bold text-gray-800 mb-2">
          Government Schemes
        </h2>
        <p className="text-gray-600 max-w-2xl mx-auto">
          Discover and apply for government agricultural schemes and subsidies
        </p>
      </div>

      <div className="bg-white rounded-2xl shadow-sm p-6">
        <div className="flex flex-col md:flex-row md:items-center justify-between mb-6">
          <h3 className="text-xl font-semibold text-gray-800 mb-4 md:mb-0">
            Available Schemes
          </h3>
          <div className="relative w-full md:w-64">
            <input
              type="text"
              placeholder="Search schemes..."
              value={searchQuery}
              onChange={(e) => setSearchQuery(e.target.value)}
              className="w-full input-field py-2 pl-10 pr-4 rounded-lg"
            />
            <i className="fas fa-search absolute left-3 top-1/2 transform -translate-y-1/2
text-gray-400"></i>
          </div>
        </div>

        {filteredSchemes.length > 0 ? (
          <div className="grid grid-cols-1 md:grid-cols-2 gap-6">
            {filteredSchemes.map((scheme, index) => (
              <div
                key={index}
                onClick={() => setSelectedScheme(scheme)}
                className="card bg-white border border-gray-200 rounded-xl p-6
cursor-pointer transition hover:border-green-500 hover:shadow-md"
              >

```

```

        <div className="flex items-start justify-between mb-4">
            <div>
                <h4 className="text-lg font-semibold
text-gray-800">{scheme.name}</h4>
                <div className="flex items-center space-x-2 mt-2">
                    <span className="bg-green-100 text-green-800 px-2 py-1
rounded-full text-xs">
                        {scheme.type}
                    </span>
                    <span className="bg-blue-100 text-blue-800 px-2 py-1
rounded-full text-xs">
                        {scheme.status}
                    </span>
                </div>
            </div>
            <div className="w-12 h-12 bg-green-100 rounded-lg flex
items-center justify-center">
                <i className={`fas fa-${scheme.icon} text-green-600`}></i>
            </div>
            <p className="text-gray-600 text-sm mb-4
line-clamp-2">{scheme.description}</p>
            <div className="flex justify-between items-center">
                <span className="text-gray-500 text-sm">Beneficiaries:
{scheme.beneficiaries}</span>
                <span className="text-green-600
font-medium">{scheme.amount}</span>
            </div>
        </div>
    )})
</div>
): (
    <div className="empty-state">
        <div className="empty-state-icon">
            <i className="fas fa-file-search text-gray-400"></i>
        </div>
        <h4 className="text-gray-700 font-medium mb-2">No schemes found</h4>
        <p className="text-gray-500 max-w-md mx-auto">
            Try adjusting your search query to find matching government schemes
        </p>
    </div>
    )}
</div>
</div>
);
};

```

// Main App Component

```

const App = () => {
  const [user, setUser] = useState(null);
  const [activeTab, setActiveTab] = useState('dashboard');

  const handleLoginSuccess = (userData) => {
    setUser(userData);
  };

  const handleLogout = () => {
    setUser(null);
    setActiveTab('dashboard');
  };

  if (!user) {
    return <LoginPage onLoginSuccess={handleLoginSuccess} />;
  }

  return (
    <div className="min-h-screen bg-gray-50">
      <Navigation activeTab={activeTab} setActiveTab={setActiveTab}
onLogout={handleLogout} />

      {activeTab === 'dashboard' && <Dashboard setActiveTab={setActiveTab} />}
      {activeTab === 'crop-disease' && <CropDiseaseDetection
setActiveTab={setActiveTab} />}
      {activeTab === 'market' && <MarketIntelligence setActiveTab={setActiveTab} />}
      {activeTab === 'voice' && <VoiceAssistant setActiveTab={setActiveTab} />}
      {activeTab === 'recommendations' && <CropRecommendations
setActiveTab={setActiveTab} />}
      {activeTab === 'schemes' && <GovernmentSchemes setActiveTab={setActiveTab}
/>}
    </div>
  );
};

// Render the app
ReactDOM.render(<App />, document.getElementById('root'));
</script> </body> </html>

```