FarmAssist - Unified Farming Management System

A comprehensive farming management system that combines crop planning, smart diagnostics, weather monitoring, financial tracking, and post-harvest management into one integrated platform.

Features

Module 1: Core Farming Engine + Daily Management

- **Login Authentication**: Secure user login to keep data tied to individual farmer profiles
- **Extension Intelligence Engine**: 100+ expert, stage-specific farming tips across 6 crops
- **AI Advisor**: Smart, context-aware advice via built-in question box
- **Growth-Stage Smart Filtering**: Auto-filters advice based on day (1–30, 31–60, etc.)
- **Priority Levels**: Color-coded tags: Critical, Important, Helpful, Optional
- **Daily Task Scheduler**: "Day 1: Clear Land" → full crop calendar automation
- **Input Calculator**: Calculates required seedlings, spacing, water, fertilizer, etc.
- **Activity Logger**: Farmers log daily activities (manually or by guided tap)
- **Offline-First PWA**: Can be used fully offline after install

Module 2: Smart Diagnostics + Voice & Accessibility Tools

- **Pest & Disease Diagnosis System**: Covers 8 crops with symptom-based decision logic
- **Image-Based Recognition**: Upload a leaf/fruit photo to detect visible crop diseases
- **Voice Accessibility**: Available in English, Yoruba, and Swahili
- **PWA Integration Ready**: Offline-compatible and modular (self-contained)
- **Modular Architecture**: Designed to plug directly into FarmAssist core
- **Documentation & Demos**: Ready for use by devs or testing by farmers/trainers

Module 3: Post-Harvest, Financials, Weather, and Photo Logs

- **Profit & Loss Tracker**: Tracks inputs and outputs per crop. Multi-currency. Calculates profit/loss per season
- **Photo Journal Timeline**: Weekly or flexible photo logging of crop progress with timestamps and notes
- **Farming Glossary**: Tap-to-expand definitions of common agri-terms. Includes placeholders for local languages
- **Environmental Monitoring**: Manual & IoT hybrid: tracks soil moisture, pH, live weather, and gives smart environmental alerts
- **Post-Harvest Tracking**: Logs batch-by-batch harvests: quantity, purpose (sold/stored/spoiled), condition, and notes
- **Weather Integration**: Real-time weather data with crop-specific advisories
- **AI Chatbot**: Intelligent farming assistant for quick questions and advice

Technology Stack

- **Frontend**: React 18, TypeScript, Tailwind CSS, Shadcn/UI
- **Backend**: Node.js, Express, TypeScript
- **Database**: PostgreSQL with Drizzle ORM
- **Authentication**: Session-based authentication
- **Weather API**: OpenWeatherMap integration
- **Build Tools**: Vite, ESBuild
- **Deployment**: Production-ready with Docker support

Installation

Prerequisites

- Node.js 18+
- PostgreSQL database
- OpenWeatherMap API key (optional, for weather features)

Setup

1. **Clone the repository**
   ```bash
   git clone <repository-url>
   cd FarmAssist-Unified
   ```

2. **Install dependencies**
   ```
   bash
   npm install
   ```

3. **Environment Configuration**
   Create a `.env` file in the root directory:
   ```
   env
   DATABASE_URL=postgresql://username:password@localhost:5432/farmassist
   OPENWEATHER_API_KEY=your_openweather_api_key
   SESSION_SECRET=your_session_secret
   ```

4. **Database Setup**
   ```
   bash
   npm run db:push
   ```

5. **Build the application**
   ```
   bash
   npm run build
   ```

6. **Start the development server**
   ```
   bash
   npm run dev
   ```

The application will be available at `http://localhost:5000`

Production Deployment

1. **Build for production**
   ```
   bash
   npm run build
   ```

2. **Start production server**
   ```
   bash
   npm start
   ```

Module Access

Core Application

- Main dashboard: `http://localhost:5000`
- Calculator: `http://localhost:5000/calculator`

- Tasks: `http://localhost:5000/dashboard`
- AI Advisor: `http://localhost:5000/advisor`

Smart Diagnostics (Module 2)

- Pest & Disease Diagnosis: `http://localhost:5000/module2/diagnostics`
- Voice Demo: `http://localhost:5000/module2/voice-demo`
- Disease Recognition: `http://localhost:5000/module2/disease-recognition`
- PWA Demo: `http://localhost:5000/module2/pwa-demo`

Extended Features (Module 3)

- Weather Dashboard: `http://localhost:5000/weather`
- Financial Tracking: `http://localhost:5000/finance`
- Photo Journal: `http://localhost:5000/photos`
- Environmental Monitoring: `http://localhost:5000/monitoring`
- Post-Harvest Tracking: `http://localhost:5000/harvest`
- Farming Glossary: `http://localhost:5000/glossary`
- AI Chatbot: `http://localhost:5000/chatbot`

Development

Available Scripts

- `npm run dev` - Start development server
- `npm run build` - Build for production
- `npm start` - Start production server
- `npm run check` - Type checking
- `npm run db:push` - Push database schema changes

Project Structure

```
FarmAssist-Unified/
├── client/              # React frontend
│   ├── src/
│   │   ├── components/    # Reusable UI components
│   │   ├── pages/         # Application pages
│   │   ├── lib/           # Utility functions
│   │   └── hooks/         # Custom React hooks
│   └── public/
│       └── module2/       # Module 2 static assets
├── server/              # Node.js backend
```

```
│   ├── routes-unified.ts   # API routes
│   ├── storage-extended.ts # Database operations
│   ├── templates/          # Module 2 HTML templates
│   └── index.ts            # Server entry point
├── shared/                 # Shared types and schemas
│   ├── schema.ts           # Main schema
│   └── schema-extended.ts  # Extended schema with Module 3
└── dist/                   # Production build output
```

Key Features Integration

Unified Navigation

All modules are accessible through a single navigation bar with intuitive icons and labels.

Shared Authentication

Single sign-on across all modules with persistent user sessions.

Integrated Database

Unified database schema supporting all module features with proper relationships.

Responsive Design

Mobile-first design that works seamlessly across desktop, tablet, and mobile devices.

Offline Capability

Progressive Web App (PWA) features for offline functionality.

Contributing

1. Fork the repository
2. Create a feature branch (`git checkout -b feature/amazing-feature`)
3. Commit your changes (`git commit -m 'Add some amazing feature'`)
4. Push to the branch (`git push origin feature/amazing-feature`)
5. Open a Pull Request

License

This project is licensed under the MIT License - see the LICENSE file for details.

Acknowledgments

- OpenWeatherMap for weather data API
- Shadcn/UI for beautiful UI components
- Lucide React for consistent iconography
- The farming community for inspiration and feedback

Support

For support, email support@farmassist.com or join our community Discord server.

FarmAssist - Empowering farmers with intelligent technology for sustainable agriculture.