# 🌾 AI Farming Advisor

An intelligent farming agent built with Google Agent Development Kit that provides data-driven crop recommendations, market analysis, and farming insights based on location, weather, soil conditions, and market prices.

## 🎯 Features

* **🌍 Location-Based Analysis**: Get recommendations based on your exact farm coordinates
* **🌤️ Real-Time Weather Data**: Current conditions, forecasts, and historical patterns
* **🌱 Soil Assessment**: Soil type, pH, nutrients, and quality analysis
* **💰 Market Intelligence**: Current crop prices, profit analysis, and market opportunities
* **📊 Crop Recommendations**: Personalized suggestions ranked by suitability and profitability
* **📅 Planting Calendar**: Optimal timing for different crops
* **🎯 Actionable Insights**: Step-by-step recommendations for farmers
* **🖥️ User-Friendly Interface**: Web-based dashboard for easy interaction

## 🏗️ Architecture

The system uses a multi-agent architecture with specialized agents:

* **Main Agent**: Orchestrates all operations using Google ADK
* **Weather Agent**: Collects and analyzes weather data
* **Soil Agent**: Processes soil conditions and compatibility
* **Market Agent**: Handles pricing and profit analysis
* **Data Processor**: Performs calculations and rankings

## 🚀 Quick Start

### Prerequisites

* Python 3.8 or higher
* Internet connection for API access

### Installation

1. **Clone or download the project:**

* cd farming\_agent

1. **Install dependencies:**

* pip install -r requirements.txt

1. **Run the web interface:**

* streamlit run ui/streamlit\_app.py

1. **Or run the command-line version:**

* python main\_agent.py --interactive

### Demo Mode

Run a quick demo with sample data:

python main\_agent.py

## 💻 Usage

### Web Interface

1. **Start the web app:**

* streamlit run ui/streamlit\_app.py

1. **Open your browser** to http://localhost:8501
2. **Enter your farm details:**
   * Latitude and longitude coordinates
   * Land area in hectares
   * Choose analysis type
3. **Get recommendations:**
   * View top crop recommendations
   * Analyze profit potential
   * Check planting calendar
   * Download results

### Command Line Interface

python main\_agent.py --interactive

Follow the prompts to enter: - Farm coordinates - Land area - Get comprehensive analysis

### Programmatic Usage

import asyncio  
from main\_agent import FarmingAgent  
  
# Initialize agent  
agent = FarmingAgent()  
  
# Get recommendations  
recommendations = asyncio.run(  
 agent.get\_comprehensive\_recommendations(  
 latitude=41.8781, # Your farm latitude  
 longitude=-93.0977, # Your farm longitude  
 land\_area=10.0 # Hectares  
 )  
)  
  
# Access results  
best\_crop = recommendations['summary']['best\_crop']  
expected\_profit = recommendations['summary']['expected\_profit']  
top\_recommendations = recommendations['top\_recommendations']

## 📊 Sample Results

The agent provides comprehensive analysis including:

### Crop Rankings

1. CORN  
 Suitability: 92.5%  
 Net profit: $4,250.00  
 ROI: 85.0%  
  
2. SOYBEANS  
 Suitability: 88.3%  
 Net profit: $3,150.00  
 ROI: 70.0%  
  
3. WHEAT  
 Suitability: 85.1%  
 Net profit: $2,800.00  
 ROI: 56.0%

### Environmental Analysis

* Current weather conditions
* Soil type and quality scores
* Seasonal forecasts
* Risk assessments

### Market Intelligence

* Current crop prices
* Best selling opportunities
* Seasonal price trends
* Market recommendations

## 🌍 Sample Locations to Try

| Region | Latitude | Longitude | Best For |
| --- | --- | --- | --- |
| Iowa, USA | 41.8781 | -93.0977 | Corn, Soybeans |
| Punjab, India | 31.1471 | 75.3412 | Wheat, Rice |
| São Paulo, Brazil | -23.5505 | -46.6333 | Various crops |
| Alberta, Canada | 53.9333 | -116.5765 | Wheat, Canola |

## 🔧 Configuration

### API Keys (Optional)

For enhanced data accuracy, you can add API keys to config.py:

WEATHER\_API\_CONFIG = {  
 'weatherapi\_key': 'YOUR\_WEATHERAPI\_KEY', # weatherapi.com  
 'ambee\_key': 'YOUR\_AMBEE\_KEY' # ambeedata.com  
}  
  
MARKET\_API\_CONFIG = {  
 'commodities\_key': 'YOUR\_COMMODITIES\_KEY' # commodities-api.com  
}

### Crop Database

Add or modify crops in config.py:

CROP\_DATABASE = {  
 'your\_crop': {  
 'optimal\_temp\_range': (15, 25), # °C  
 'rainfall\_requirement': (400, 600), # mm/year  
 'soil\_ph\_range': (6.0, 7.0),  
 'growing\_season\_days': 90,  
 'soil\_types': ['loam', 'sandy loam'],  
 'planting\_months': [4, 5, 6]  
 }  
}

## 📂 Project Structure

farming\_agent/  
├── main\_agent.py # Main orchestrating agent  
├── config.py # Configuration and crop database  
├── requirements.txt # Dependencies  
├── README.md # This file  
├── agents/ # Specialized agents  
│ ├── weather\_agent.py  
│ ├── soil\_agent.py  
│ └── market\_agent.py  
├── utils/ # Utilities  
│ └── data\_processor.py  
└── ui/ # User interfaces  
 └── streamlit\_app.py

## 🧪 Testing

Test the system with different locations:

# Test different climates  
locations = [  
 (41.8781, -93.0977), # Temperate (Iowa)  
 (31.1471, 75.3412), # Subtropical (Punjab)  
 (-23.5505, -46.6333) # Tropical (São Paulo)  
]  
  
for lat, lon in locations:  
 recommendations = asyncio.run(  
 agent.get\_comprehensive\_recommendations(lat, lon, 5.0)  
 )  
 print(f"Best crop for {lat}, {lon}: {recommendations['summary']['best\_crop']}")

## 🔍 How It Works

1. **Location Analysis**: Takes farm coordinates and analyzes geographical patterns
2. **Environmental Data**: Collects real-time weather, soil, and climate data
3. **Crop Matching**: Compares conditions against crop requirements database
4. **Market Analysis**: Evaluates current prices and profit potential
5. **Scoring System**: Ranks crops using weighted scoring (60% suitability, 40% profit)
6. **Risk Assessment**: Evaluates weather, market, and agricultural risks
7. **Recommendations**: Provides actionable insights and next steps

## 🎯 Use Cases

* **Small Farmers**: Get recommendations for family farms
* **Commercial Operations**: Analyze large-scale crop decisions
* **Agricultural Consultants**: Provide data-driven advice to clients
* **Research**: Study crop suitability patterns across regions
* **Education**: Learn about agricultural decision-making

## 🌟 Key Benefits

* **Data-Driven Decisions**: Remove guesswork from crop selection
* **Profit Optimization**: Focus on most profitable opportunities
* **Risk Reduction**: Understand and mitigate agricultural risks
* **Timing Optimization**: Plant at the right time for best results
* **Market Intelligence**: Stay informed about pricing trends
* **Easy to Use**: No technical expertise required

## 🔧 Customization

### Adding New Crops

1. Add crop data to config.py:

'new\_crop': {  
 'optimal\_temp\_range': (min\_temp, max\_temp),  
 'rainfall\_requirement': (min\_rain, max\_rain),  
 'soil\_ph\_range': (min\_ph, max\_ph),  
 'growing\_season\_days': days,  
 'soil\_types': ['soil\_type1', 'soil\_type2'],  
 'planting\_months': [month1, month2]  
}

1. Add pricing data to market\_agent.py:

self.base\_prices['new\_crop'] = price\_per\_ton  
self.production\_costs['new\_crop'] = cost\_per\_hectare  
self.typical\_yields['new\_crop'] = tons\_per\_hectare

### Regional Adaptation

Modify regional price factors in config.py:

REGIONAL\_PRICE\_FACTORS = {  
 'your\_region': multiplier  
}

## 🐛 Troubleshooting

**Common Issues:**

1. **Import Errors**: Ensure all dependencies are installed

* pip install -r requirements.txt

1. **API Timeouts**: Check internet connection and API endpoints
2. **Location Errors**: Verify latitude/longitude format (decimal degrees)
3. **Streamlit Issues**: Update Streamlit to latest version

* pip install --upgrade streamlit

## 🤝 Contributing

Contributions welcome! Areas for improvement:

* Additional data sources
* More sophisticated ML models
* Enhanced risk assessment
* Mobile interface
* Offline capabilities
* Multi-language support

## 📄 License

Open source - feel free to use and modify for your projects.

## 🙏 Acknowledgments

* Google Agent Development Kit team
* Open weather data providers
* Agricultural research community
* Farming communities worldwide

**Happy Farming! 🌾**

For questions or support, please check the documentation or create an issue.