

AI-Powered Agricultural Market Intelligence & Supply Chain Platform

Project Documentation

1. Title and Scope

Title:

AI-Powered Agricultural Market Intelligence & Supply Chain Platform

Scope:

This project is designed to transform traditional agricultural market operations through real-time crop price prediction, multilingual FAQ support, and smart logistics integration. The system empowers farmers, distributors, and aggregators with data insights for better pricing, decision-making, and supply chain management.

2. Purpose / Objective

- Predict real-time crop prices using AI/ML models
- Analyze market demand and supply trends regionally
- Provide multilingual FAQ/chatbot support to farmers
- Automate alerts on price changes and weather conditions
-
- Optimize inventory and logistics routes through partner APIs
- Offer role-based dashboards for different stakeholders

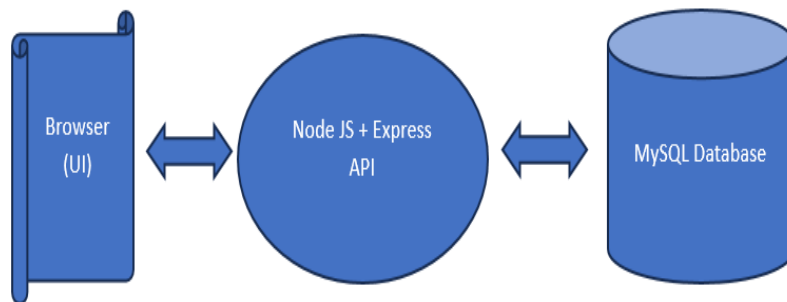
3. Project Overview

A backend application built using **Node.js** and **Express.js**, exposing REST APIs for real-time crop data, user management, chat-based FAQs, and integration with logistics systems. It uses a **MySQL** or **MongoDB** database and supports a **React.js** frontend.

4. Architecture Overview

System Architecture Diagram:

A monolithic REST API architecture with modular routes, services, and integrations.



Architecture Type:

Monolithic REST API server

- **Technical Stack**

Component	Technology
Frontend	React.js, Tailwind CSS
Backend	Node.js, Express.js
Database	MySQL / MongoDB
AI Models	Python (scikit-learn, TensorFlow)
Authentication	JWT (JSON Web Token)
Hosting	AWS / Render / Heroku / Vercel
Version Control	Git + GitHub
Category	Technologies / Tools
AI/ML Frameworks	TensorFlow, PyTorch
Data Processing	Apache Spark, Pandas
Speech Processing	Google Speech-to-Text, Amazon Transcribe
APIs & Integration	RESTful APIs, n8n, Make.com
Frontend	React.js, Angular
Backend	Node.js, Python (FastAPI or Django)
Database	PostgreSQL, MongoDB

Cloud Platforms	AWS, Azure, GCP
Data Visualization	Power BI, Tableau

5. Modules & Components

Module	Responsibilities
Authentication	Login, register, JWT, role-based access
Crop Price Engine	Fetch and predict crop prices from government APIs
FAQ Chatbot	Multilingual question-answer module using NLP
Demand Forecasting	AI model to predict crop demand based on location & time
User Management	Create, update, and assign roles to users
Logistics Integration	API-based inventory and route optimization
Dashboard	Visual reports and insights for farmers, aggregators, distributors

Component Integration:

Express routers for modular structure, middleware for auth/logging, services for DB interaction and AI.

6. Data Design (High-Level)

ER Diagram Summary:

Users ↔ Crops ↔ Predictions

Users ↔ FAQs

Users ↔ Logistics ↔ Inventory

Tables (Example - SQL):

- **users:** id, name, role, email, password
- **crops:** id, name, region, price, unit

- **faq:** id, language, question, answer
 - **logistics:** id, user_id, route, inventory_level, vehicle_id
 - **predictions:** id, crop_id, forecast_price, date
-

7. Integration and Interfaces

API Interface:

- REST API over HTTP
- JSON format for request/response

Sample APIs:

POST /api/login

GET /api/crops

POST /api/predict

GET /api/faqs?lang=mr

POST /api/logistics/optimize

8. Security Considerations

- JWT for authentication
 - Password hashing via bcrypt
 - Input validation via middleware
 - Role-based access (Admin, Farmer, Aggregator, Distributor)
 - API rate limiting and CORS
-

9. Risk Factors

- Data mismatch from external APIs
- Forecast inaccuracies during weather anomalies
- Latency in logistics API responses
- Language model limitations in regional dialects
- Server downtime without redundancy

10. Deployment Architecture

Server:

Cloud server (AWS EC2/Render)

Deployment Process:

- Use PM2 or Docker for backend deployment
- Nginx as reverse proxy
- Continuous Deployment via GitHub Actions or Vercel (for frontend)

11. Low-Level Design

Example Class-like Module:

```
// predictionModel.js
{
  crop_id,
  forecast_price,
  region,
  predicted_on
}
```

12. Directory Structure

```
/routes    → Express route handlers
/controllers → Controller logic
/services  → Business logic + DB queries
/models    → Schema or data structures
/config    → Env variables, DB configs
/ai        → ML/AI related models
/utils     → Helpers like validators
```

13. Detailed Database Design

Main Tables:

1. users
2. crops
3. price
4. predictions
5. faq
6. logistics
7. routes
8. inventory

14. API Design Module-Wise

Endpoint	Method	Description
/api/login	POST	Authenticate user
/api/register	POST	Register new user
/api/crops	GET	List available crops
/api/predict	POST	Predict price of crop
/api/faqs	GET	Fetch FAQs in selected language
/api/logistics/optimize	POST	Optimize logistics route

15. Third-Party Libraries

- express – Server framework
- mysql2 / mongoose – DB clients
- bcryptjs – Password hashing
- jsonwebtoken – JWT auth
- dotenv – Env variable support
- axios – API calls

- cors, helmet – Security

16. Exception Handling

- Centralized error handler middleware
 - Log to file or cloud service (e.g., LogRocket, Sentry)
 - Validate all incoming data
 - Try-catch around DB/API calls
-

17. Coding Standards

- ESLint + Prettier for formatting
 - kebab-case for files, camelCase for variables
 - Services and controllers named properly
 - API versioning using /api/v1/
-

18. Naming Conventions

Item	Convention
REST APIs	/api/resource-name
Service Files	cropService.js
Controller	cropController.js
DB Tables	snake_case (e.g., crop_data)

19. Tech Stack / Tools

Category	Technologies / Tools
AI/ML Frameworks	TensorFlow, PyTorch
Data Processing	Apache Spark, Pandas
Speech Processing	Google Speech-to-Text, Amazon Transcribe
APIs & Integration	RESTful APIs, n8n (workflow automation), Make.com

Category	Technologies / Tools
Frontend	React.js, Angular
Backend	Node.js, Python (FastAPI or Django)
Database	PostgreSQL, MongoDB
Cloud Platforms	Amazon Web Services (AWS), Microsoft Azure, Google Cloud Platform (GCP)
Data Visualization	Power BI, Tableau

20. Database Design

1. users

Field Name	Data Type	Constraints	Description
id	INT	PK, AUTO_INCREMENT	Unique user ID
name	VARCHAR(100)	NOT NULL	User’s full name
email	VARCHAR(100)	NOT NULL, UNIQUE	Email ID for login
password	VARCHAR(255)	NOT NULL	Hashed password
role	ENUM	DEFAULT 'farmer'	farmer/aggregator/distributor/admin
language	VARCHAR(10)	DEFAULT 'en'	Preferred language code
created_at	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP	Registration date

2. crops

Field Name	Data Type	Constraints	Description
id	INT	PK, AUTO_INCREMENT	Unique crop ID
name	VARCHAR(100)	NOT NULL	Name of the crop
region	VARCHAR(100)	NOT NULL	Region where crop is cultivated
unit	VARCHAR(50)	NOT NULL	Measurement unit (kg/ton/litre)

created_at	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP	Added date
------------	-----------	------------------------------	------------

3. Price

Field Name	Data Type	Constraints	Description
id	INT	PK, AUTO_INCREMENT	Unique price entry ID
crop_id	INT	FK → crops(id)	Crop being tracked
region	VARCHAR(100)	NOT NULL	Region of pricing
price	DECIMAL(10,2)	NOT NULL	Price in INR
date	DATE	NOT NULL	Date of price record
source	VARCHAR(100)	NULLABLE	Source of price data (API/manual)

3. predictions

Field Name	Data Type	Constraints	Description
id	INT	PK, AUTO_INCREMENT	Unique prediction ID
crop_id	INT	FK → crops(id)	Crop for which prediction is generated
forecast_price	DECIMAL(10,2)	NOT NULL	Predicted price
predicted_on	DATE	DEFAULT CURRENT_DATE	Date of prediction
accuracy	DECIMAL(5,2)	NULLABLE	Model accuracy

4. faq (frequently asked questions)

Field Name	Data Type	Constraints	Description
id	INT	PK, AUTO_INCREMENT	Unique FAQ ID
language	VARCHAR(10)	NOT NULL	Language code (e.g., en, mr, hi)
question	TEXT	NOT NULL	User question
answer	TEXT	NOT NULL	Chatbot answer

5. logistics

Field Name	Data Type	Constraints	Description
id	INT	PK, AUTO_INCREMENT	Unique logistics ID
user_id	INT	FK → users(id)	Associated distributor/aggregator
crop_id	INT	FK → crops(id)	Crop being transported
origin	VARCHAR(100)	NOT NULL	Pickup location
destination	VARCHAR(100)	NOT NULL	Delivery location
status	ENUM	DEFAULT 'pending'	pending / in_transit / delivered
vehicle_id	VARCHAR(50)	NULLABLE	Assigned vehicle number
updated_at	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP	Last update

6. routes

Field Name	Data Type	Constraints	Description
id	INT	PK, AUTO_INCREMENT	Unique route ID
origin	VARCHAR(100)	NOT NULL	Start point
destination	VARCHAR(100)	NOT NULL	End point
distance_km	DECIMAL(6,2)	NOT NULL	Distance in kilometers
optimal_time	VARCHAR(50)	NULLABLE	Estimated time
road_type	VARCHAR(50)	NULLABLE	Highway, village road, etc.

7. inventory

Field Name	Data Type	Constraints	Description
id	INT	PK, AUTO_INCREMENT	Unique inventory ID
user_id	INT	FK → users(id)	Owner of the inventory
crop_id	INT	FK → crops(id)	Crop in stock
quantity	DECIMAL(10,2)	NOT NULL	Stock quantity
unit	VARCHAR(20)	NOT NULL	Measurement unit
last_updated	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP	Last stock update