

# Demeter - Product Requirements Document

## AI-Powered Digital Twin for Smallholder Maize Production

**Version:** 1.0

**Date:** February 25, 2026

**Team:** 3-4 members (Frontend, Backend/Java, AI Engineer, Hardware)

---

### 1. Executive Summary

**Demeter** is an AI-powered Digital Twin platform that helps smallholder maize farmers in Sub-Saharan Africa predict climate risks and simulate farming decisions before committing resources.

#### One-Liner

An AI farm co-pilot that lets farmers test “what-if” scenarios and receive SMS-based decision guidance.

#### Key Value Proposition

- Predict drought stress 7-14 days ahead
  - Simulate scenarios (dry spells, delayed planting, irrigation changes)
  - Alert farmers via SMS with actionable recommendations
- 

### 2. Problem Statement

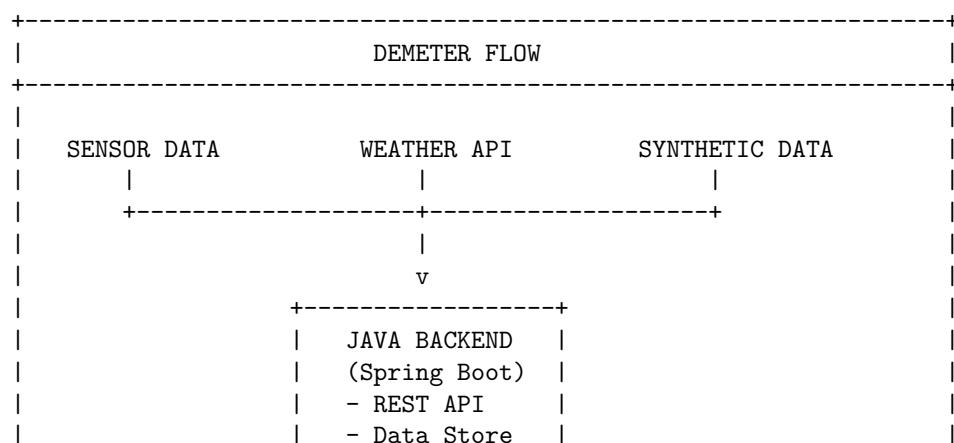
- Sub-Saharan Africa loses **\$4B+** annually in maize production due to climate volatility
- Smallholder farmers (80% of food production) make decisions **blind to near-term risks**
- A single misjudged decision during dry spell = **30-50% yield loss**

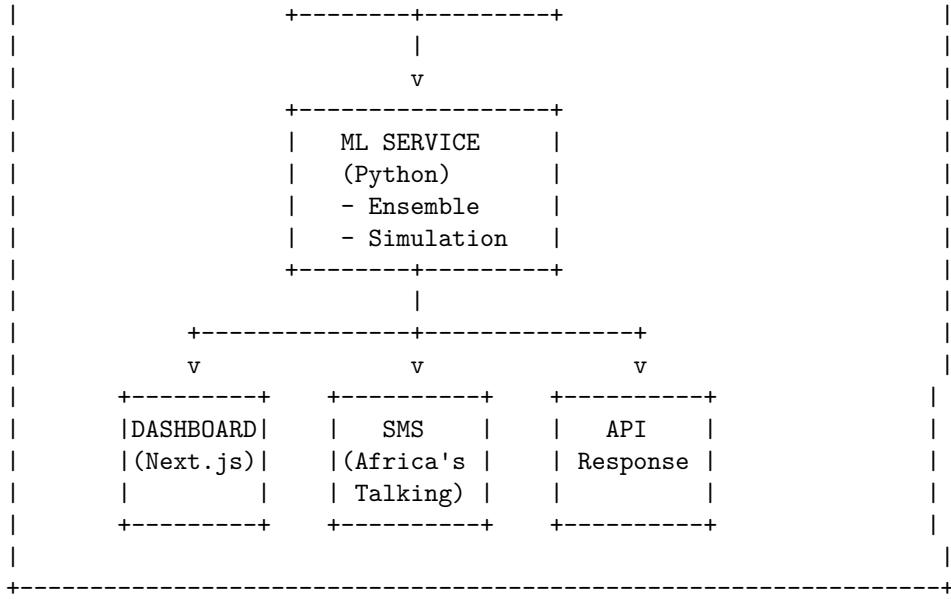
#### Target Persona

**Amina**, 38, Kaduna State, Nigeria - Grows maize on 1.5 hectares - Feature phone, uses mobile money - Rain-fed agriculture - Lost 45% of harvest last season to unexpected dry spell

---

### 3. Solution Overview

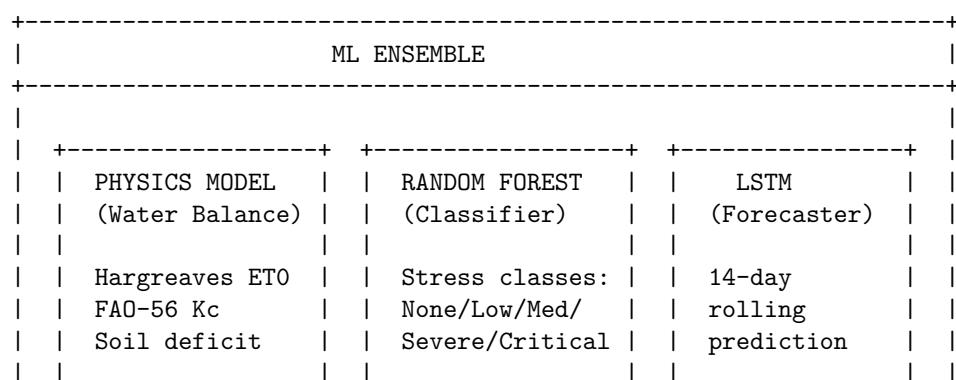


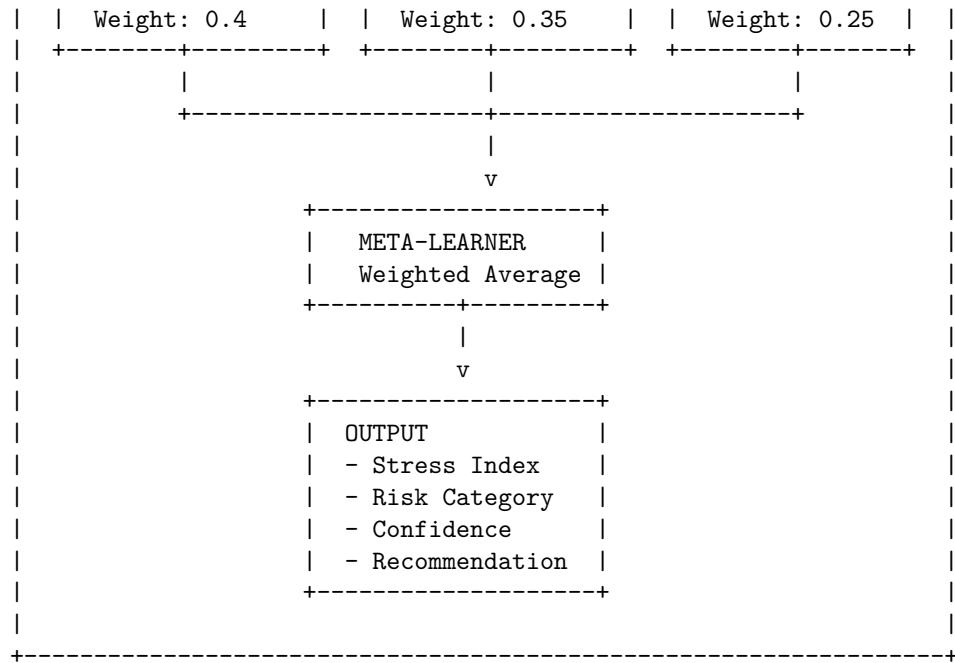


#### 4. Core Features

ID	Feature	Priority	Owner
F1	Sensor data ingestion (real + synthetic)	P0	Backend + Hardware
F2	Weather data integration (Open-Meteo)	P0	Backend
F3	ML Ensemble prediction engine	P0	AI Engineer
F4	Scenario simulation API	P0	AI Engineer + Backend
F5	Dashboard with risk visualization	P0	Frontend
F6	SMS alerts via Africa's Talking	P1	Backend
F7	Farm management (CRUD)	P1	Backend + Frontend

#### 5. ML Ensemble Architecture





## Model Inputs

- Soil moisture (%)
- Temperature (C)
- Humidity (%)
- Rainfall (mm) - actual + forecast
- Crop growth stage
- Days since last rain

## Model Outputs

- stress\_index: 0-100
- risk\_category: None | Low | Moderate | Severe | Critical
- confidence: 0-1
- days\_to\_critical: integer
- recommendation: string

## 6. Tech Stack

Layer	Technology	Owner
<b>Frontend</b>	Next.js 14 + TypeScript + Tailwind + Recharts	Frontend Dev
<b>Backend API</b>	Java 17 + Spring Boot 3	Backend Dev
<b>ML Service</b>	Python 3.11 + FastAPI + scikit-learn + TensorFlow	AI Engineer
<b>Database</b>	PostgreSQL	Backend Dev
<b>Hardware</b>	ESP32 + DHT22 + Soil Moisture Sensor	Hardware Engineer
<b>Weather API</b>	Open-Meteo (free)	Backend Dev
<b>SMS</b>	Africa's Talking	Backend Dev
<b>Hosting</b>	Vercel (FE) + Railway (BE + ML)	All

---

## 7. API Contracts

### 7.1 Sensor Data Ingestion

```
POST /api/v1/sensor-data
{
  "farm_id": "uuid",
  "soil_moisture": 45.2,
  "temperature": 32.5,
  "humidity": 65.0,
  "timestamp": "2026-02-25T10:30:00Z"
}
```

### 7.2 Get Prediction

```
GET /api/v1/farms/{farm_id}/prediction
```

Response:

```
{
  "farm_id": "uuid",
  "stress_index": 72,
  "risk_category": "SEVERE",
  "confidence": 0.85,
  "days_to_critical": 4,
  "recommendation": "Irrigate 25mm within 3 days",
  "forecast": [
    {"day": 1, "stress": 72},
    {"day": 2, "stress": 76},
    ...
  ]
}
```

### 7.3 Run Simulation

```
POST /api/v1/farms/{farm_id}/simulate
{
  "scenario": "DRY_WEEK",
  "parameters": {
    "duration_days": 14,
    "rainfall_mm": 0
  }
}
```

Response:

```
{
  "baseline": {"stress_index": 45, "yield_impact": 0},
  "simulated": {"stress_index": 82, "yield_impact": -35},
  "recommendation": "Irrigate 25mm before day 3 to avoid critical stress"
}
```

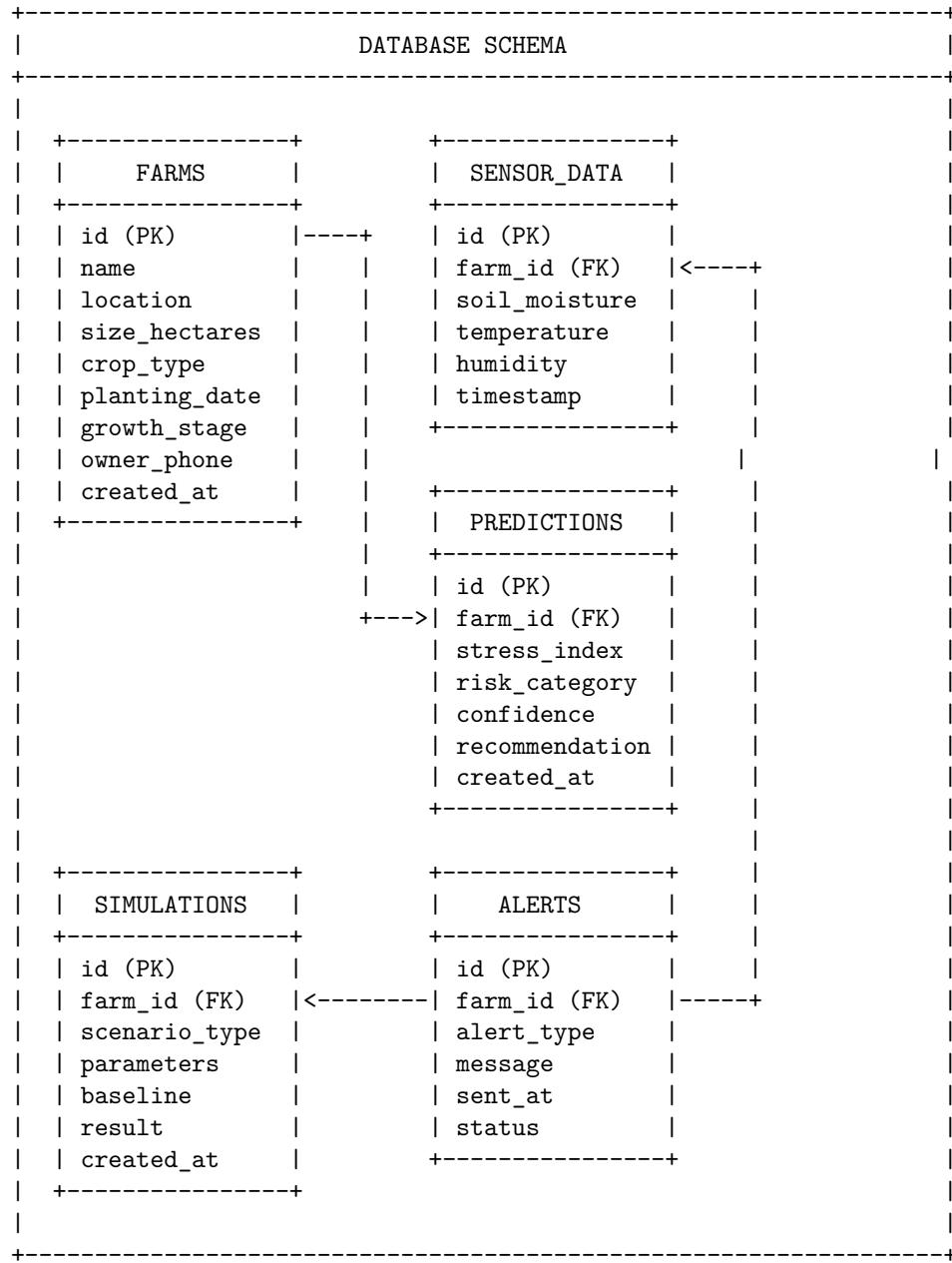
### 7.4 Send SMS Alert

```
POST /api/v1/alerts/sms
```

```
{
  "farm_id": "uuid",
  "phone": "+2348012345678",
  "language": "en" // or "ha" for Hausa
}
```

---

## 8. Database Schema




---

## 9. Simulation Scenarios

Scenario	Parameters	Use Case
DRY_WEEK	duration_days, rainfall_mm=0	“What if no rain for 14 days?”
DELAYED_PLANTING	delay_days	“What if I plant 2 weeks late?”
IRRIGATION_TEST	irrigation_mm	“What if I irrigate 20mm now?”
CUSTOM	Any overrides	Free-form scenario

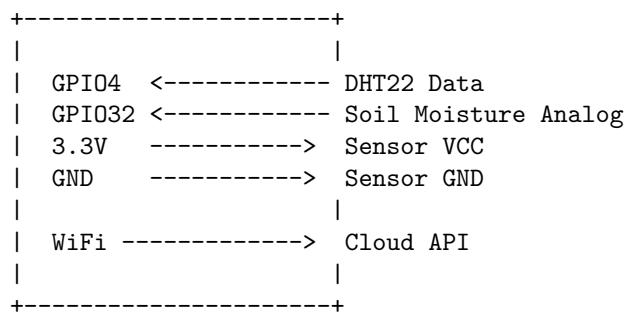
## 10. Hardware Specification

### Components

- **MCU:** ESP32 DevKit (~\$5)
- **Temp/Humidity:** DHT22 (~\$3)
- **Soil Moisture:** Capacitive sensor (~\$2)
- **Power:** 5V USB or solar panel

### Wiring Diagram

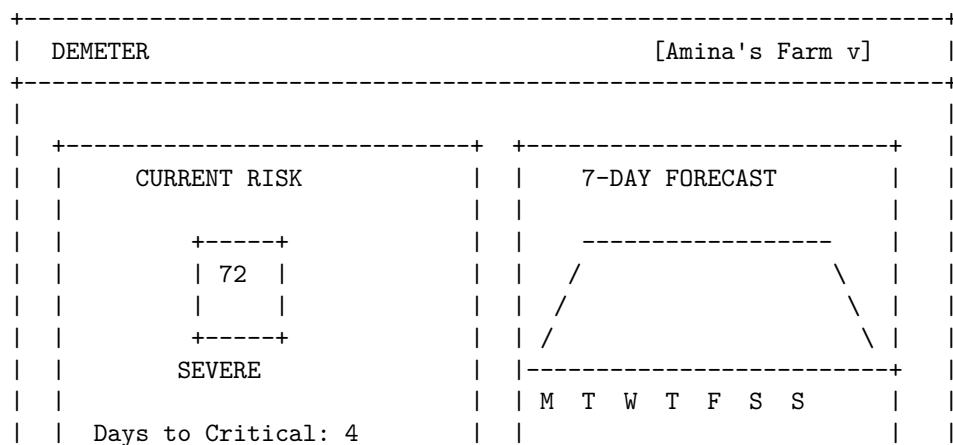
ESP32 DevKit



### Data Transmission

- Read sensors every **5 minutes**
- POST to `/api/v1/sensor-data`
- Retry with exponential backoff on failure

## 11. Dashboard Wireframe



SENSOR DATA	SIMULATION
Soil Moisture: 32%	[Dry Week v]
Temperature: 34C	
Humidity: 45%	Duration: [14] days
Last Rain: 5 days ago	[Run Simulation]

RECOMMENDATION
! Irrigate 25mm within 3 days to avoid 35% yield loss
[Send SMS Alert]

## 12. Team Responsibilities

Member	Role	Deliverables
Frontend Dev	Dashboard	Next.js app, charts, simulation UI
Backend Dev (Java)	API + Integration	Spring Boot API, DB, Africa's Talking, Open-Meteo
AI Engineer	ML Service	Python FastAPI, ensemble models, simulation engine
Hardware Engineer	IoT	ESP32 firmware, synthetic data generator

## 13. Demo Script (5-7 min)

- Hook (30s):** “Amina lost 45% of her maize last season. What if she could have seen it coming?”
- Dashboard Tour (1 min):** Show current farm health, sensor data
- Simulation Demo (2 min):**
  - Click “Simulate Dry Week (14 days)”
  - Watch stress index climb from 45 to 82
  - Show “Critical in 4 days” warning
- SMS Alert (30s):** Send alert, show message on phone
- Architecture (30s):** Quick diagram slide
- Scale Story (1 min):** “One farm to Platform to Insurance data”
- Close (30s):** “Demeter gives farmers foresight. Help us scale it.”

## 14. Success Metrics

Metric	Target
Sensor to Prediction latency	< 5 seconds
Prediction accuracy (vs FAO baseline)	+/- 15%
Simulation response time	< 2 seconds
SMS delivery success	> 95%

---

## 15. External Services

Service	Purpose	Signup
<b>Open-Meteo</b>	Weather data	No API key needed
<b>Africa's Talking</b>	SMS	<a href="https://africastalking.com">https://africastalking.com</a> (free sandbox)
<b>Railway</b>	Backend hosting	<a href="https://railway.app">https://railway.app</a>
<b>Vercel</b>	Frontend hosting	<a href="https://vercel.com">https://vercel.com</a>

---

## 16. Risk Mitigation

Risk	Mitigation
Hardware fails during demo	Synthetic data generator as fallback
ML model inaccurate	Physics model provides baseline guarantee
API rate limits	Cache weather data, batch requests
SMS not delivered	Show SMS in dashboard as backup

---

## 17. Project Structure

```
demeter/
|-- README.md
|-- PRD.md
|-- backend/                      # Java Spring Boot
|   |-- src/main/java/
|   |-- pom.xml
|   +-- ...
|-- ml-service/                   # Python FastAPI
|   |-- app/
|   |   |-- main.py
|   |   |-- models/
|   |       |-- water_balance.py
|   |       |-- random_forest.py
|   |       |-- lstm.py
|   |       +-- ensemble.py
|   |-- simulation/
|   +-- synthetic/
|-- requirements.txt
```

```
|    +-- ...
|-- frontend/                 # Next.js
|    |-- src/
|    |-- package.json
|    +-- ...
+-- hardware/                 # ESP32 Arduino
    |-- demeter_sensor/
    |    +-- demeter_sensor.ino
    +-- ...
```

---

Let's build this and win!