Demeter - Frontend Engineer PRD

Dashboard & User Interface

Role: Frontend Developer
Stack: Next.js 14 + TypeScript + Tailwind CSS + Recharts

---

1. Overview

Build a real-time dashboard that displays farm health, sensor data, risk predictions, and allows farmers/extension officers to run "what-if" simulations.

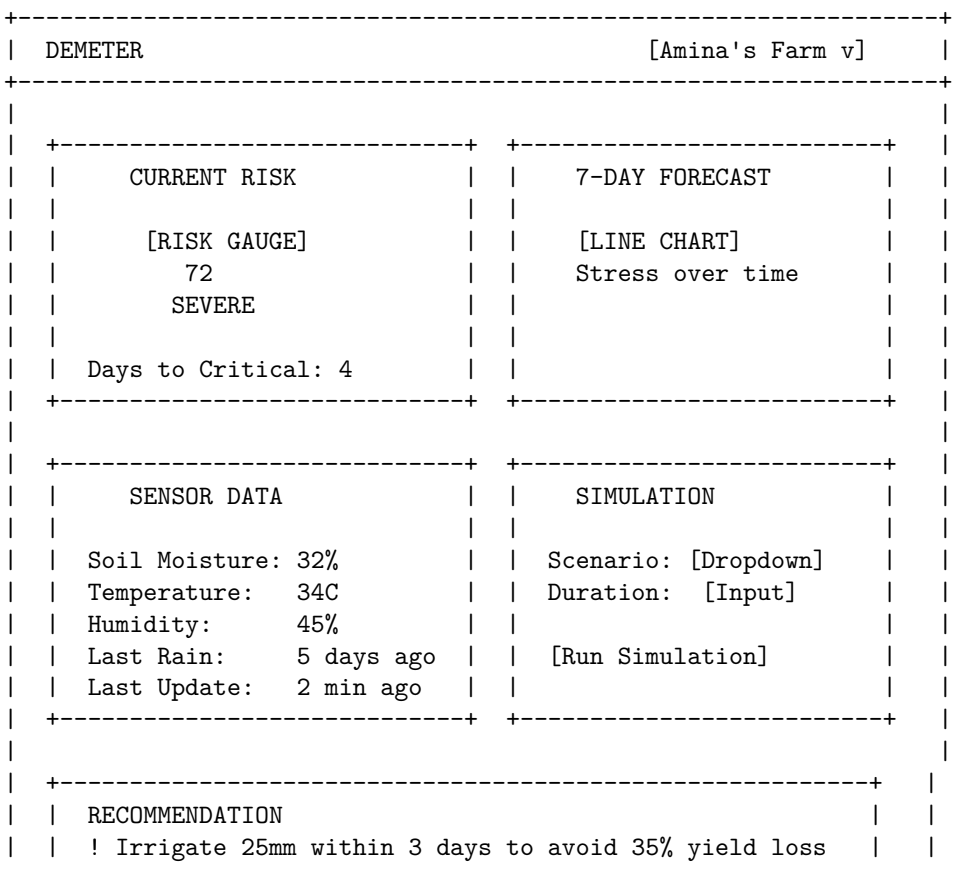
---

2. Pages & Routes

Route	Page	Description
`/`	Landing	Hero + login/register
`/dashboard`	Main Dashboard	Risk gauge, charts, simulation panel
`/farms`	Farm List	View/manage multiple farms
`/farms/[id]`	Farm Detail	Single farm deep dive
`/farms/new`	Add Farm	Create new farm
`/settings`	Settings	SMS preferences, profile

---

3. Dashboard Layout



```
|  |  [Send SMS Alert]    [View History]                      |   |
|  +--------------------------------------------------------+   |
+---------------------------------------------------------------+
```

---

4. Components to Build

4.1 Core Components

Component	Props	Description
`RiskGauge`	`value: number, category: string`	Circular gauge 0-100 with color coding
`StressChart`	`data: {day, stress}[]`	Line chart showing 7-14 day forecast
`SensorCard`	`label, value, unit, trend`	Display single sensor reading
`SimulationPanel`	`onSimulate: (params) => void`	Scenario selector + parameters + button
`RecommendationBanner`	`message, severity`	Alert banner with action buttons
`FarmSelector`	`farms: Farm[], selected: string`	Dropdown to switch farms

4.2 Form Components

Component	Description
`FarmForm`	Create/edit farm (name, location, size, planting date)
`AlertSettings`	SMS number, language preference, alert thresholds

4.3 Layout Components

Component	Description
`Navbar`	Logo, farm selector, user menu
`Sidebar`	Navigation links (optional)
`DashboardGrid`	Responsive grid layout

---

5. API Integration

5.1 Endpoints to Call

```typescript
// Types
interface Farm {
  id: string;
  name: string;
  location: string;
  size_hectares: number;
  crop_type: string;
  planting_date: string;
  growth_stage: string;
  owner_phone: string;
}
```

```typescript
interface SensorData {
  soil_moisture: number;
  temperature: number;
  humidity: number;
  timestamp: string;
}

interface Prediction {
  stress_index: number;
  risk_category: 'NONE' | 'LOW' | 'MODERATE' | 'SEVERE' | 'CRITICAL';
  confidence: number;
  days_to_critical: number;
  recommendation: string;
  forecast: { day: number; stress: number }[];
}

interface SimulationResult {
  baseline: { stress_index: number; yield_impact: number };
  simulated: { stress_index: number; yield_impact: number };
  recommendation: string;
}
```

5.2 API Calls

```
// Fetch farm prediction
GET /api/v1/farms/{farmId}/prediction
Response: Prediction

// Fetch latest sensor data
GET /api/v1/farms/{farmId}/sensor-data/latest
Response: SensorData

// Run simulation
POST /api/v1/farms/{farmId}/simulate
Body: { scenario: string, parameters: object }
Response: SimulationResult

// Send SMS alert
POST /api/v1/alerts/sms
Body: { farm_id: string, phone: string, language: string }
Response: { success: boolean, message_id: string }

// CRUD Farms
GET    /api/v1/farms
POST   /api/v1/farms
GET    /api/v1/farms/{id}
PUT    /api/v1/farms/{id}
DELETE /api/v1/farms/{id}
```

---

6. State Management

Use React Query (TanStack Query) for server state:

```typescript
// hooks/useFarmPrediction.ts
export function useFarmPrediction(farmId: string) {
  return useQuery({
```

```
    queryKey: ['prediction', farmId],
    queryFn: () => fetchPrediction(farmId),
    refetchInterval: 30000, // Refresh every 30s
  });
}

// hooks/useSimulation.ts
export function useSimulation(farmId: string) {
  return useMutation({
    mutationFn: (params: SimulationParams) => runSimulation(farmId, params),
  });
}
```

---

7. Risk Color Coding

Risk Category	Color	Stress Range
NONE	Green (#22c55e)	0-20
LOW	Lime (#84cc16)	21-40
MODERATE	Yellow (#eab308)	41-60
SEVERE	Orange (#f97316)	61-80
CRITICAL	Red (#ef4444)	81-100

---

8. Simulation Panel UX

Scenario Options

```
const scenarios = [
  { value: 'DRY_WEEK', label: 'Simulate Dry Period' },
  { value: 'DELAYED_PLANTING', label: 'Delayed Planting' },
  { value: 'IRRIGATION_TEST', label: 'Test Irrigation' },
  { value: 'CUSTOM', label: 'Custom Scenario' },
];
```

Parameter Inputs by Scenario

Scenario	Parameters
DRY_WEEK	duration_days (slider 1-21)
DELAYED_PLANTING	delay_days (slider 1-30)
IRRIGATION_TEST	irrigation_mm (slider 5-50)
CUSTOM	rainfall_mm, irrigation_mm, duration_days

Results Display

Show side-by-side comparison:

```
+------------------------+------------------------+
|       BASELINE         |      SIMULATED         |
|                        |                        |
|   Stress: 45           |   Stress: 82           |
|   Yield Impact: 0%     |   Yield Impact: -35%   |
|                        |                        |
+------------------------+------------------------+
```

---

9. Responsive Design

Breakpoint	Layout
Mobile (<640px)	Single column, stacked cards
Tablet (640-1024px)	2 columns
Desktop (>1024px)	Full grid as wireframe

---

10. Project Structure

```
frontend/
||| src/
|   ||| app/
|   |   ||| layout.tsx
|   |   ||| page.tsx                # Landing
|   |   ||| dashboard/
|   |   |   ||| page.tsx
|   |   ||| farms/
|   |   |   ||| page.tsx            # Farm list
|   |   |   ||| new/page.tsx        # Add farm
|   |   |   ||| [id]/page.tsx       # Farm detail
|   |   ||| settings/
|   |       ||| page.tsx
|   ||| components/
|   |   ||| ui/                     # Shadcn/ui components
|   |   ||| RiskGauge.tsx
|   |   ||| StressChart.tsx
|   |   ||| SensorCard.tsx
|   |   ||| SimulationPanel.tsx
|   |   ||| RecommendationBanner.tsx
|   |   ||| FarmSelector.tsx
|   |   ||| Navbar.tsx
|   ||| hooks/
|   |   ||| useFarmPrediction.ts
|   |   ||| useSensorData.ts
|   |   ||| useSimulation.ts
|   |   ||| useFarms.ts
|   ||| lib/
|   |   ||| api.ts                  # API client
|   |   ||| utils.ts
|   ||| types/
|       ||| index.ts                # TypeScript types
||| public/
||| tailwind.config.js
||| next.config.js
||| package.json
||| tsconfig.json
```

---

11. Dependencies

```
{
  "dependencies": {
    "next": "^14.0.0",
    "react": "^18.0.0",
```

```
    "react-dom": "^18.0.0",
    "@tanstack/react-query": "^5.0.0",
    "recharts": "^2.10.0",
    "tailwindcss": "^3.4.0",
    "class-variance-authority": "^0.7.0",
    "clsx": "^2.0.0",
    "lucide-react": "^0.300.0"
  }
}
```

---

12. Environment Variables

```
NEXT_PUBLIC_API_URL=http://localhost:8080/api/v1
```

---

13. Deliverables Checklist

- ☐ Project setup (Next.js + Tailwind + TypeScript)
- ☐ Landing page
- ☐ Dashboard layout with responsive grid
- ☐ RiskGauge component
- ☐ StressChart component (Recharts)
- ☐ SensorCard components
- ☐ SimulationPanel with scenario dropdown
- ☐ Simulation results comparison view
- ☐ RecommendationBanner with SMS button
- ☐ FarmSelector dropdown
- ☐ Farm CRUD pages
- ☐ API integration hooks
- ☐ Loading & error states
- ☐ Mobile responsive design

---

14. Demo Requirements

For the hackathon demo, ensure: 1. Dashboard loads with realistic data (mock if needed) 2. Simulation button triggers visible change in risk gauge 3. SMS button shows confirmation toast 4. Smooth animations on gauge/chart updates

---

Coordinate with: Backend (API contracts), AI Engineer (prediction format)