🔷 SECTION 1 — Product Context (Give This First)

This is a production-ready executive kiosk application for a joint IBM–Microsoft event in Bangalore on 8 March 2026.

Purpose:

To guide customers through scenario-based questions and dynamically recommend the most relevant IBM–Microsoft offering.

Offerings included:

1. Data Transformation with Microsoft

2. Application Migration & Modernization (AMM)

3. Digital Product Design & Engineering (DPDE)

The application must:

- Be executive-grade and visually polished

- Work on large touchscreen displays

- Provide fast (<3 minute) guided interaction

- Dynamically calculate recommendation

- Display a rich recommendation screen with solution highlights, assets, demos, and QR codes

---

🔷 SECTION 2 — Functional Requirements

SCREEN FLOW:

Screen 1: Welcome

- IBM + Microsoft co-branded header

- Event title: IBM–Microsoft Global Event 2026

- CTA: “Start Your Transformation Journey”

Screen 2: Main Scenario Selection

- Show all scenarios given per offering

- User selects one scenario

Screen 3: Sub-Scenario (Single Select)

- 3–4 options as per pdf files in doc folder
- Each has:
    - short description
    - signal path
    - weight

Screens 4–8: Questions (5 total)

- Fixed question set per main scenario
- 4 options per question
- Each option:
    - signalPath
    - weight

Screen 9: Recommendation

- Primary offering
- Supporting offering (if threshold met)
- Optional capability (if threshold met)
- Show:
    - Key challenges
    - IBM + Microsoft solution
    - Delivery approach (keep this as common to all offering recommendations) you are free to modify as per IBM consulting standard
    - Key capabilities
    - Technology stack
    - Demos / Architecture / Tools (here just a placeholder for now)
    - QR code to download report

🔷 SECTION 3 — Scoring Engine Requirements

You must explicitly define this so the logic is not misinterpreted.

SCORING MODEL:

Each selection produces:

- signalPath
- weight

Each signalPath maps to weighted contributions across offerings.

Example:

Signal: Data Platform Modernization

Mapping:

Data +3

AI +1

AMM 0

DPDE 0

For each selection:

offeringScore += weight × offeringMultiplier

After all selections:

Sort offering scores descending.

Primary Offering = highest score

Supporting = second highest if >= 40% of primary

Optional = third highest if >= 25% of primary

Also specify:

Scoring must be:

- Deterministic
- Transparent
- Easy to adjust via JSON configuration

🔷 SECTION 4 — Data Model Requirement

Tell Claude explicitly:

All content (scenarios, questions, mappings, recommendation content) must be stored in structured JSON configuration files.

No hard-coded logic inside components.

Structure:

```
{
 scenarios: [
  {
   id,
   title,
   description,
   offeringGroup,
   subScenarios: [],
   questions: [],
   signalPathMappings: []
  }
 ],
 signalMappingMatrix: []
}
```

Also require:

- Modular architecture
- Separate config per offering
- Central scoring engine module
- UI components reusable

🔷 SECTION 5 — UI & Production Requirements

Design Requirements:

- Modern Microsoft Fabric aesthetic
- Clean IBM enterprise style
- Soft gradients
- Large typography for touchscreen
- Card-based selection UI
- Animated progress bar
- Executive dashboard feel

Performance:

- Must load instantly
- No API dependencies required
- Fully local config-based logic

Deployment:

- Should be deployable via:
  - Static hosting (Azure Static Web Apps)
  - Or containerized React app

Code Stack:

- React + TypeScript
- Tailwind CSS
- Framer Motion animations
- State managed via context or Zustand

🏗️ What NOT To Do

Do NOT:

- Hardcode scenario logic into components
- Embed scoring logic inside UI layer
- Mix content with logic

- Build without modular architecture

🧠 Strategic Tip

This must be architected like an enterprise-grade configurable decision engine, not a simple form app.

That sentence changes everything.