Here's a **streamlined assignment** for your candidates, focusing on **database design, AI prompting, and dashboard creation** without prescribing specific hosting solutions. This allows candidates to choose their preferred tools while ensuring you can review their work.

Subject: Assignment: Recycling Production Line Manager Selection System

Hi [Candidate Name],

Thank you for your interest in **G CP**! Your assignment is to build a **minimal, standalone system** for ranking candidates for a **recycling production line manager** role. Focus on **database design, AI prompting, and a functional dashboard**.

📌 Requirements

1. **Database Design:**
    - Create a **MySQL-compatible schema** for candidates, evaluations, and rankings.
    - Include tables for:
        - `candidates` (ID, name, experience, skills).
        - `evaluations` (AI scores for crisis management, sustainability, team motivation).
        - `rankings` (auto-updated based on evaluations).
    - Share the **SQL schema file** and a **sample dataset** (40 candidates).
2. **Random Candidate Generator:**
    - Generate **40 realistic candidate profiles** using Faker.js.
    - Include fields for experience, skills, and random attributes.
3. **AI Prompting:**
    - Create **3 prompts** to evaluate candidates on:
        - Crisis management.
        - Sustainability knowledge.
        - Team motivation.
    - Use **mock AI responses** or a free AI API (e.g., Claude, Copilot).
4. **Dashboard:**
    - Build a **React + Vite dashboard** with:
        - Leaderboard (top 10 candidates).
        - Skill heatmap (visualize scores).
        - Candidate cards (detailed profiles).

    - Use Mantine UI for components.

5. **Submission:**

    - Provide a **GitHub repository** with:
        - Frontend code (React + Vite).
        - SQL schema and sample data.
        - AI prompts (Markdown/PDF).
    - Include a **README** with setup instructions.

📤 Submission Guidelines

Reply with:

1. **GitHub Repository** (code + SQL schema).
2. **AI Prompts** (Markdown/PDF).
3. **Screenshot/Video** of the dashboard.

Deadline: [Insert date]

🌟 Evaluation Criteria

Area	Weight	What We Look For
Database Design	30%	Schema efficiency, indexes, triggers.
AI Prompting	30%	Creativity, depth, rubric clarity.
Dashboard	20%	Usability, visual clarity.
Random Data	20%	Realism, code quality.

💡 Tips

- Use **Mantine's** `<Table>` for sortable/filterable tables.
- For AI mocks, return random scores if no API is used.
- **Bonus:** Add a "Share Candidate" button to simulate HR workflows.
