Hackathon Project Brief – Part 1 (Pre-Hack Requirements)

SDG 8: AI-Powered Youth Employment & Career Roadmap Platform

Context

This document contains the pre-hack (Part 1) requirements for the hackathon project. Part 2 (onsite challenges and AI requirements) will be revealed on the hackathon day.

Theme & Problem

Many students and unemployed youth struggle to identify relevant job opportunities, understand their existing skills, and plan their learning journey. Your solution should support SDG 8: Promote sustained, inclusive and sustainable economic growth, full and productive employment and decent work for all, by helping young people connect their skills to real opportunities.

High-Level Concept

Build a fullstack web application where users (students, fresh graduates, job seekers) can create a profile, manage their skills, and discover suitable jobs and learning resources. In Part 1, you will focus on a clean foundation: authentication, profile, data models, and basic (non-AI) recommendation logic.

General Technical Guidelines

You can pick any technology/tech-stacks.

- Fullstack web app using any modern stack.
- Persistent database for users, jobs, and learning resources.
- Clean, responsive, and usable UI.
- Deployed or easily runnable locally with clear instructions.

PART 1 – REQUIRED FEATURES (TO BE COMPLETED BEFORE ONSITE)

1. Authentication & User Management

- Implement user registration and login (email/password or similar secure method).
- Basic validation (e.g., required fields, invalid email format, password length).

- Store at minimum:
    - Full name
    - Email
    - Education level / department
    - Experience level (e.g., Fresher / Junior / Mid)
    - Preferred career track (e.g., Web Development, Data, Design, Marketing, etc.).

2. User Profile & Skill Input

- Create a Profile page for each user.
- Users must be able to:
    - Add or edit a list of skills (e.g., “JavaScript”, “Communication”, “Graphic Design”).
    - Add brief experience or project descriptions.
    - Indicate career interests or target roles.
- Additionally:
    - Provide a simple way to paste CV text or notes for later analysis (no AI/processing is required in Part 1; just store this data).

3. Jobs & Opportunities Database (Seeded Data)

- Create and seed a jobs/opportunities collection or table.
- Each job entry should include at least:
    - Job Title
    - Company/Organization
    - Location or “Remote”
    - Required skills (array/list)
    - Recommended experience level
    - Job type (Internship / Part-time / Full-time / Freelance)
- Include a minimum of 15–20 relevant entries, focused on student and entry-level roles.
- Build a simple Jobs page with:
    - List view of jobs
    - Basic filter options (e.g., by role/track, location, type)
    - Job details view.

4. Learning Resources / Courses Collection

- Create and seed a learning resources collection or table.
- Each resource should include:
    - Title
    - Platform (e.g., YouTube, Coursera, Udemy, local platforms)
    - URL
    - Related skill(s)
    - Cost indicator (Free / Paid)

- Include at least 15–20 resources mapped to common skills (e.g., HTML, JS, Excel, communication, design).
- Show these resources on a dedicated page or section.

5. Basic Matching Logic (Non-AI)

- Implement a simple rule-based recommendation feature:
  - Use the user's selected skills and preferred track.
  - Recommend jobs where there is reasonable skill overlap.
  - Show why a job is recommended (e.g., "Matches: JavaScript, HTML").
- Similarly, recommend learning resources relevant to the user's existing or desired skills.
- - This logic can be simple but must be consistent and transparent.

6. User Dashboard & UI

- Create a dashboard/home view for logged-in users that includes:
  - Quick view of profile and skills.
  - Recommended jobs (from your basic matching).
  - Recommended learning resources.
- Ensure navigation is clear (e.g., Navbar with: Dashboard, Jobs, Resources, Profile, Logout).
- The design should prioritize clarity, accessibility, and real usability over heavy visuals.

7. Documentation & Code Quality

- Include a README in your repository with:
  - Project overview (2–3 lines).
  - Tech stack used.
  - Setup steps (how to install dependencies and run frontend/backend).
  - Any environment configuration notes.
  - Seed data usage instructions.
- Code should be organized logically (separate routes/controllers/models where applicable).

What Teams Must Bring to the Onsite Hackathon

- A working fullstack application implementing all Part 1 requirements.
- Code pushed to a public or shareable repository.
- Ability to run the project on a new machine using the README.

Important Notes

- No advanced AI features are required in Part 1.

- However, design your data structures and flows so that AI and smarter logic can be integrated in Part 2 without rewriting everything.
- Teams that come with an incomplete Part 1 base will be at a disadvantage during the onsite hackathon.