Full Stack Developer Assignment – CodeAtRandom AI

Goal of This Assignment:

Build a skill-gap analysis, career roadmap, and public API integration.

Features to Build

1. **Career Goal Input Page-**

A simple page where the user can enter:

a. **Target Role** (text input, e.g., "Backend Developer")

b. **Current Skills** (multi-select or comma-separated input)

c. Click → **Analyze My Career Path**

2. **Skill Gap Analyzer API (POST /api/skill-gap)**

Create an API: ***POST /api/skill-gap***

a. Uses predefined JSON

```
{
"Frontend Developer": ["HTML", "CSS", "JavaScript", "React", "Git"],
"Backend Developer": ["Java", "Spring Boot", "SQL", "APIs", "Git"],
"Data Analyst": ["Excel", "SQL", "Python", "Dashboards", "Statistics"]
}
```

b. Output JSON

- matched skills
- missing skills
- recommendations
- suggested learning order

This should be returned via API, not frontend logic.

3. **Career Roadmap Generator API (POST /api/roadmap)**

Logic (Mock AI)
Based on target role:
Return a 3-level roadmap:
Example: Backend Developer

- Phase 1 (1–2 months): Java basics, OOP, Git
- Phase 2 (2 months): Spring Boot, SQL, APIs
- Phase 3 (1–2 months): Deployment, projects, system design basics

4. **Public API Integration (HackerNews API)**

Task: Integrate **HackerNews API** (no key required)
Fetch: Top 5 latest tech stories
API: https://github.com/HackerNews/API
Then fetch details using: https://hacker-news.firebaseio.com/v0/item/<id>.json
Frontend Should Display:

- Title
- URL
- Score
- Time
- Type
- By

Assignment

5. **Combined Dashboard Page**

After analysis, show:

- Left side → Skill Gap Results
- Right side → Career Roadmap
- Bottom section → Latest Tech News (from HackerNews API)

Layout requirement:

- neat, minimal, responsive
- does NOT need to look pretty
- should be structured & usable

Tech Requirements

- Frontend: React or Next.js (preferred)
- Backend: Node.js/Express or Python (FastAPI/Flask)
- Database: Optional (But bonus if candidate uses JSON → Aile storage to save user inputs)

Deliverables

1. GitHub repository
2. README with setup and API description
3. Live hosted link on vercel or any other plan.

General Instructions:

1. **Expected Effort & Time**
   - This assignment is designed to be completed in **7-8 hours**.
   - Do not aim for perfection — aim for clarity, structure, and functionality.
   - You are free to take more time if needed, but we value **smart prioritization**.

2. **Technology Choices**

You may use the following:

Frontend (choose any):

- React
- Next.js

Backend (choose any):

- Node.js + Express
- Python (FastAPI or Flask)

Database:

- Not required for this round
- You MAY use JSON or in-memory storage if needed

Your stack choice will not affect evaluation — clarity and logic matter more.

3. **Allowed Modifications**

You may make the following **nominal changes**:

- You may slightly modify UI layouts as long as all outputs are clearly visible.
- You may structure backend routes differently **as long as required APIs exist**.
- You may add additional fields or improve the UI for usability.
- You may add comments, helper functions, or modular folder structures.

Assignment

Do NOT:

- Change the core requirement of skill-gap API, roadmap API, or HackerNews integration.
- Use AI to generate roadmap dynamically (mock logic only).

4. Use of Libraries

Allowed:

- Any UI component library (Material UI, Tailwind, Chakra, etc.)
- Axios / Fetch for API calls
- Any state management (Context, Redux, Zustand, etc.)
- Utility libraries like Lodash, Moment, etc.

Not allowed:

- No-code tools
- Pre-built full AI roadmap generators
- Complete template projects copied from GitHub

We want **your** implementation.

5. Code Quality Expectations

- Clean folder structure
- Meaningful function and variable names
- Clear separation of frontend & backend code
- Proper error handling (especially in API calls)
- Write comments where logic may not be obvious
- Keep components and API handlers small and readable

Your code does not have to be perfect — it should simply show **your thought process and engineering approach**.

6. Deliverables

Submit:

i) GitHub Repository (Public or Private Link)

- /frontendfolder
- /backendfolder
- Instructions to run both

ii) README (Important!)

Include:

- Tech stack used
- How to run frontend
- How to run backend
- API endpoints
- Any assumptions

iii) Live Hosted link on vercel or netlify or any other:

This helps us understand your decision-making.

7. Evaluation Criteria

We evaluate on:

- **Correctness**: Do the APIs and UI work as expected?
- **API Handling**: Can you integrate a public API & custom backend APIs cleanly?
- **Product Understanding**: Does your logic align with career guidance?
- **Code Quality**: Clean structure, readable files, maintainability
- **UI/UX**: Simple, clear, and easy to use
- **Problem Solving**: How you structure logic like skill matching & roadmaps

Assignment

- **Communication**: README and optional video clarity.

This is NOT a UI design test.
This is a **full-stack thinking + execution** test.

8. **Communication & Support**
    - You may Google or refer to documentation.
    - You may NOT ask others to do the assignment.
    - If something is unclear, make reasonable assumptions and document them in the README.

    We value **clarity of thought**, not guessing.

Assignment