Junior Full-Stack Developer Technical Screening – I

Environment Pollutants in Chandigarh (2019-2021)

Time to Complete

You will have 4 days from the time you receive this assignment to complete and submit your solution.

Objective

Build a secure Environment Pollutants Dashboard with a MongoDB Atlas DB, NodeJS backend and a Next.js frontend, deployed on Render (backend) and Vercel (frontend).

You are free to choose your hosting provider (AWS, Azure, Vercel, Firebase, Render, Replit etc)

The dashboard should:

- Require login with JWT authentication.
- Allow CRUD operations on dataset.
- Provide filtering by year, year range, and month range.
- Display a line graph showing Benzene, Toluene, and NO emissions per month for a given year.

Requirements

1. Database (MongoDB Atlas) - https://www.mongodb.com/

1.1 Create a free MongoDB Atlas cluster, then name your DB and add a collection named **"opengds"**

1.2 Define a schema with fields: Use the link below to download the **raw dataset in JSON format**. **This RAW data may need processing and cleaning depending on your schema, so please take care of this additional step. In case of link failure feel free to ask on WhatsApp Group.**

https://www.data.gov.in/backend/dms/v1/ogdp/resource/download/602920807/json/eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpc3MiOiJkYXRhLmdvdi5pbiIsImF1ZCI6ImRhdGEuZ292LmluIiwiaWF0IjoxNzU2MTIyNzI1LCJleHAiOjE3NTYxMjMwMjUsImRhdGEiOnsibmlkIjoiNjAyOTIwODA3In19.OSx_CEbdO2ONNOoRZ9pd3hj7fho4dzJF_aStr4h4WeA

1.3 Insert all records of the dataset into your collection named "**opengds**"

2. Backend (Node.js + Express)

Setup a new NodeJS project and create the following routes.

1.1 Authentication (JWT)

- POST /api/auth/signup → Register user (store hashed password).
- POST /api/auth/login → Login with email & password → return JWT.

All /api/data/* routes must require a valid JWT.

Data APIs

- POST /api/data → Add new record (auth required).
- /api/data?year=2021
  a. Default filter → returns data for **2021** if no filter applied.
  b. If **no month selected**, return **all months**.
- /api/data?startYear=2019&endYear=2021
  a. Year range filter → return monthly data for all years in range.
- /api/data?year=2020&startMonth=1&endMonth=6
  a. Year + month range filter → return **Jan–Jun 2020**.

- PUT /api/data/:id → Edit record (auth required).
- DELETE /api/data/:id → Delete record (auth required).

Additional API: Pollution Trend Graph

GET /api/data/trends?year=2020 → Return dataset for line graph.

Response example:

```
{
 "year": "2020",
 "monthlyTrends": [
  {"month": "Jan", "Benzene": 3.1, "Toluene": 1.5, "NO": 17.2},
  {"month": "Feb", "Benzene": 2.8, "Toluene": 1.3, "NO": 15.9},
  {"month": "Mar", "Benzene": 3.9, "Toluene": 1.7, "NO": 18.1}
 ]
```

}

*This will be used on the front end to render three-line graphs **(Benzene, Toluene, NO)** across months.*

3. Frontend (Next.js + shadcn/ui)

- Use Next.js (App Router).
- Use shadcn/ui components for forms, tables, filters, and modals.
- Login/Register pages → JWT auth flow.
- Dashboard (protected route):
- Show paginated table of records.
- **Implement three filters**:
    a. Single Year (default 2021) → Show full year, unless months are selected.
    b. Year Range (e.g., 2019–2021).
    c. Year + Month Range (e.g., Jan–Jun 2020).
- Add Add/Edit/Delete functionality.
- Add pagination controls for table
- Add line graph view using data from /api/data/trends.

4. Deployment

- Deploy backend (Express app) on Render.
- Deploy frontend (Next.js app) on Vercel.
- Ensure frontend APIs point to the deployed backend.

5. Deliverables

Public GitHub repo links:

- **Backend (Express + JWT + MongoDB Atlas)**
- **Frontend (Next.js + shadcn + JWT auth flow).**
- **Deployed links (Render + Vercel).**

Bonus (Optional but Appreciated)

- Add charts (Recharts / Chart.js) with animations.
- Add toast notifications and loading states – react skeleton,cliploaders etc.

- The Graph data and Table Data should instantly refresh after mutation/updation - Use React Query if familiar.
- Use **Next Auth** for Authentication (Optional)
- Pagination on Frontend and Backend