Add cover

DE Technical Challenge

👋 Welcome!

Thank you for taking the time to complete this technical challenge. We're excited to get to know you through your code, your thinking, and how you approach data engineering problems.

This exercise is not just about building features — it's about understanding your **engineering mindset**, **design skills**, and how you communicate your decisions.

This technical challenge is a chance for you to **show us how you think, build, and structure your work.**

> **Estimated time: 4 hours, you can use more if needed but, remember, we are not looking for the complete and perfect solution.**
>
> **Goal: Show us your strengths — not perfection!**
>
> **Don't worry if you can't finish everything. Do what you can and make it yours. You will have the technical interview to explain more on your approach.**

We know your time is valuable, and we truly appreciate your interest in joining our Data & AI Engineering team. Let's dive in!

🧬 The Challenge: "Clinical Trial Data Pipeline"

Your goal is to build a data pipeline for a life sciences consultancy that processes clinical trial data from multiple sources.

The pipeline should allow users to ingest data from various formats, process and validate it, store it efficiently, and provide analytical insights about patient outcomes and trial performance.

This is a first step toward a more complex production system — so we're also curious to see how you would design for **scalability**, use **best practices**, and include **proper documentation**.

✅ Functional Requirements

Your prototype should support:

- **Uploading data from multiple sources**: CSV files, JSON APIs, SQL databases
- **Data processing and validation**: Handle missing values, standardize formats, validate integrity
- **Efficient storage**: Design appropriate data models for different data types
- **Analytics and reporting**: Generate insights about patient enrollment, adverse events, treatment efficacy

---

🛠️ Technical Expectations

We intentionally leave architectural choices open — we want to understand your reasoning. However, we're especially interested in seeing how you approach:

- **Python development**: clean, maintainable code with proper structure
- **SQL proficiency**: data manipulation, analytical queries, optimization

- **Docker containerization**: environment management and deployment readiness
- **Data pipeline design**: ETL/ELT patterns, orchestration, error handling
- **Testing and validation**: unit tests, data quality checks, edge case handling
- **Documentation**: clear explanations and setup instructions

You're welcome to use any databases (PostgreSQL, SQLite, etc.), data processing libraries (pandas, polars, etc.), and workflow tools (Airflow, Prefect, etc.).

You can also add a **web interface** with Streamlit, FastAPI, or any framework of your choice.

📊 Evaluation Focus

We will assess your submission across the following dimensions:

	Area	Focus
1	**1. Functionality**	Core features implemented and working
2	**2. Code Quality**	Readability, structure, naming, documentation
3	**3. Data Engineering Concepts**	Understanding of ETL/ELT, data modeling, pipeline design
4	**4. SQL Proficiency**	Query optimization, data manipulation, analytical queries
5	**5. Architecture & Design**	Modularity, scalability considerations, technical decision-making
6	**6. Docker & DevOps**	Containerization, environment management
7	**7. Testing & Quality**	Test coverage, validation logic, error handling
8	**8. Documentation**	Clear explanations, setup instructions, design decisions
9	**9. Execution**	Ease of running the project locally as described in the README

📋 Public Dataset

We'll be using a **publicly available clinical trial dataset** for this challenge. **Part of your challenge is to design an appropriate database schema** to store this data efficiently and support the required analytics.

Verified Dataset Sources

Choose **ONE** of the following verified public datasets:

Option 1: Kaggle Clinical Trials Dataset 🔗 ClinicalTrials.gov Dataset

- Recent data from June 2024
- Multiple CSV files with study information, interventions, outcomes
- Requires free Kaggle account to download

Option 2: Alternative Clinical Trials 🔗 All Clinical Trials Dataset

- Comprehensive clinical trials data
- Single CSV file, easier to start with
- Free Kaggle account required

Option 3: COVID-19 Clinical Trials 🔗 COVID-19 Clinical Trials

- Focused dataset on COVID-19 trials
- Well-structured with clear relationships
- Good for demonstrating domain knowledge

Option 4: Use ClinicalTrials.gov API (Advanced)

- Access live data via API: `https://clinicaltrials.gov/api/v2/studies`
- Perfect for candidates who want to show API integration skills
- Documentation: Search "ClinicalTrials.gov API v2" for current docs

Your Database Design Challenge

Working with your chosen dataset, you need to:

- **Explore and understand** the structure of the data
- **Design a normalized database schema** that efficiently stores this data
- **Define appropriate relationships** between entities (studies, interventions, outcomes, etc.)
- **Choose suitable data types** and constraints
- **Consider indexing strategies** for performance
- **Handle data quality issues** you discover in the data
- **Document your design decisions** in your README

Expected Analytics

Your pipeline should be able to answer questions like:

- How many trials by study type and phase?
- What are the most common conditions being studied?
- Which interventions have the highest completion rates?
- Geographic distribution of clinical trials
- Timeline analysis of study durations

Note: All these datasets contain real-world inconsistencies, missing values, and formatting variations - exactly what you'd encounter in production!

🤖 AI Tools & Coding Assistance

We're totally cool with you using AI tools! Whether it's GitHub Copilot, ChatGPT, Claude, or any other coding assistant - use whatever helps you be productive. Modern development includes these tools, and we want to see how you work with them.

What we care about:

- **You understand the code**: Be ready to explain every part of your solution
- **Technical decisions are yours**: We want to see your reasoning for model choices, architecture decisions, and trade-offs
- **Transparency is key**: If you used AI assistance heavily, mention it in your README
- **Share your prompts**: If you have interesting prompts that guided your AI assistant, feel free to include them - we love seeing effective prompt engineering!

Bottom line: Use AI to accelerate your work, but make sure you can own and explain the entire solution during our discussion.

📤 Submission Instructions

When finished, please follow these steps to submit your solution:

1. **Push your code to a public GitHub repository**

(Or private repo + invite reviewer)

2. **Ensure the README.md includes:** • Project overview and architecture diagram (can be simple)

• Clear local setup instructions (including Docker)

• Reasoning behind design decisions

• Trade-offs and limitations

• Time allocation breakdown

• Future improvements and scalability considerations

3. **If using a private repo, please grant access to the following GitHub account(s):**

➤ `MIGx-user`

4. **Submit your GitHub link via email.**

🧐 Bonus Questions (Answer in README)

Please provide brief answers to these questions in your README:

1. **Scalability**: How would you modify your solution to handle 100x more data volume?
2. **Data Quality**: What additional data validation rules would you implement for clinical trial data?
3. **Compliance**: If this were a GxP environment, what additional considerations would you need?
4. **Monitoring**: How would you monitor this pipeline in production?
5. **Security**: What security measures would you implement for sensitive clinical data?

💡 Tips for Success

- **Start Simple**: Get a basic pipeline working first, then add complexity
- **Document as You Go**: Don't leave documentation to the end
- **Test Early**: Write tests for your core functions
- **Git Best Practices**: Make meaningful commits with clear messages
- **Time Management**: Spend roughly 1 hour on setup/architecture, 2 hours on implementation, 1 hour on testing/documentation

---

🙏 Thank You

We truly appreciate the time and effort you're putting into this.

We're looking forward to reviewing your solution and discussing it further with you during the interview.

Good luck — and have fun! 🚀