🎯 Witty Data Interview Assessment

Role: Mid–Senior Software Engineer / App Developer (PERN Stack)

📌 Project Overview

You are required to build a simple internal web application to manage NDIS Rate Sets and Invoices using the PERN stack (PostgreSQL, Express/Koa, React, Node.js).

This project assesses your full stack development capabilities including backend validation, frontend forms, data modeling, Excel file parsing, and user interface design.

🛠️ Tech Stack

- PostgreSQL
- Node.js (Koa)
- React + Ant Design
- XLSX (Excel file parsing)
- **dayjs** for date handling
- **bignumber.js** for monetary calculations

⏳ Time Estimate

Estimated Time to Complete: 12–18 hours

(You can work in modular parts — it doesn't need to be continuous.)

📁 Modules to Build

1. Rate Set Module
   1.1. List all available NDIS rate sets
   1.2. Add new NDIS rate sets and import corresponding data from Excel
   1.3. Update existing rate sets and view associated data
2. Invoice Module
   2.1. List all existing invoices
   2.2. Add new invoices
   2.3. Edit existing invoices and invoice items

🗃️ Data Models (Summary)

Each model must follow strict validation rules and include appropriate timestamp tracking.

- NDIS Rate Set
- Support Category
- Support Item
- Support Item Value
- Invoice
- Invoice Item

Refer to the detailed specification for field names, Excel column mapping, validation logic, and relationships.

✅ Key Validation Rules

- **Names** must be unique, not whitespace-only
- **Dates** must not:
  - Overlap with existing rate sets
  - Leave a gap in between
  - Have invalid Start/End logic
- **Invoices** can have multiple Rate Sets, but each **invoice item** must match exactly one Rate Set
- **Monetary Values** must be handled with **bignumber.js** and:

- Allow up to 2 decimal places
- Use thousand separators (e.g. 1,234,567.89)

📦 Deliverables

- Frontend code (named **my-ndis-app**)
- Backend code (named **my-ndis-api**)
- SQL schema / migration script
- README including:
  - Setup instructions
  - How to import Excel files
  - Any assumptions made
- (Optional) Sample DB with seed data
- (Optional) Postman collection or Swagger docs

Invoicing Logic

Each **invoice item** must:

- Match a valid Rate Set for the given date range
- Pull Support Categories and Items from the matched Rate Set
- Use:

  Invoiced Amount = Unit × Invoiced Rate

- Respect maximum rate by region from Support Item Value
- Ensure:

  Sum of **Invoice Items** = Expected Invoiced Amount

🧪 Evaluation Criteria

Category	Weight
Code quality & structure	✅✅✅
Schema design & validation	✅✅✅
React UI + Form validation	✅✅
Excel import parsing	✅✅
Matching logic for invoices	✅✅✅
Correct use of libraries	✅✅
Documentation & clarity	✅✅
Git usage	✅

🎁 Bonus / Stretch Goals

- Add authentication (JWT or session-based)
- Containerize and deploy the app using Docker/Docker Compose
- Add unit or integration tests
- Add pagination, search, filtering in lists
- Use clean Git commits and branches
- Optimize Excel import for large datasets

📝 Notes

- All **arithmetic with money** must use **bignumber.js**.
- All dates should be timezone-aware (**timestamp with time zone**).
- Excel file fields must match sample format (columns A to AB).
- If two rate sets match a given invoice item range → show error.
- If no rate set matches → show error.

🔍 Detail Specification

NDIS Rate Set

Field	Validation Rules
Name	Required Must be unique Not whitespace-only
Start Date	Required Overlap with existing rate sets Leave a gap in between Have invalid Start/End logic
End Date	Required Overlap with existing rate sets Leave a gap in between Have invalid Start/End logic
Enabled	Default: true

Support Category

Field	Excel Mapping	Validation Rules
Category Number	**F –** Support Category Number (PACE)	Required Not whitespace-only
Category Name	**H –** Support Category Name (PACE)	Required Not whitespace-only
Sorting *1		Default: 0

***1 – Convert Category Number to integer as Sorting, Order by Sorting**

Support Item

Field	Excel Mapping	Validation Rules
Item Number	**A –** Support Item Number (PACE)	Required Not whitespace-only
Item Name	**B –** Support Item Name (PACE)	Required Not whitespace-only
Unit	**I** – Unit	
Quote	**J –** Quote	
Start Date	**K –** Start Date	
End Date	**L –** End Date	
Non-Face-to-Face Support Provision	**W –** Non-Face-to-Face Support Provision	
Provider Travel	**X –** Provider Travel	
Short Notice Cancellations	**Y –** Short Notice Cancellations.	
NDIA Requested Reports	**Z –** NDIA Requested Reports	
Irregular SIL Supports	**AA –** Irregular SIL Supports	
Type	**AB –** Type	
Sorting *1		Default: 0

***1 – Convert Category Number to integer as Sorting, Order by Sorting**

Support Item Value

Field	Excel Mapping	Validation Rules
Code	**M1 – V1:** Use the cell value as Code (ACT, NSW, NT, QLD, SA, TAS, VIC, WA, Remote, Very Remote)	Required Not whitespace-only
Label	**M1 – V1:** Use the cell value as Label (ACT, NSW, NT, QLD, SA, TAS, VIC, WA, Remote, Very Remote)	Required Not whitespace-only
Value	**M – V**	
Sorting	ACT = 1 NSW = 2 NT = 3 QLD = 4 SA = 5	Default: 0

	TAS = 6 VIC = 7 WA = 8 Remote = 9 Very Remote = 10	

Invoice

Field	**Validation Rules**
Invoice Date	Required
Invoice Number	Required Must be unique Not whitespace-only
Expected Invoiced Amount	Required Allow up to 2 decimal places Use thousand separators (e.g. 1,234,567.89)

Invoice Item

Field Name	**Description**	**Validation Rules**
Service Start Date	Once selected, must match exactly one Rate Set	Required Have invalid Start/End logic No future dates
Service End Date	Once selected, must match exactly one Rate Set	Required Have invalid Start/End logic No future dates
Support Category	List all support categories of the matching NDIS rate set based on selected dates	Required
Support Item	List all support items based on selected Support Category	Required
Rate Region	List all support item values based on selected Support Item	Required
Max Rate	Use the “Value” of the selected Rate Region as Max Rate	Read-only Allow up to 2 decimal places Use thousand separators (e.g. 1,234,567.89)

Unit	To calculate the Invoiced Amount: **Unit x Invoiced Rate = Invoiced Amount**	Required Allow up to 2 decimal places Use thousand separators (e.g. 1,234,567.89)
Invoiced Rate	To calculate the Invoiced Amount: **Unit x Invoiced Rate = Invoiced Amount**	Required Allow up to 2 decimal places Use thousand separators (e.g. 1,234,567.89)
Invoiced Amount	**Unit x Invoiced Rate = Invoiced Amount**	Read-only Allow up to 2 decimal places Use thousand separators (e.g. 1,234,567.89)