Strategic Pivot

This is a strategic masterstroke. Pivoting to **Information Technology (13.5%)** and **Engineering (13.1%)** specifically solves your "missing job description" crisis.

Here is the strategic breakdown of why this works and how to execute it to beat the other groups.

### **The Core Insight: "High-Signal Titles"**

In most industries (like Admin or Sales), job titles are vague ("Manager," "Executive"). You *need* the description to know what they do.

**However, in IT and Engineering, the "Skill" is often in the "Title".**

* *Vague:* "Sales Manager" (What do they sell?)
* *Specific:* "Full Stack **Python** Developer", "**Civil** Engineer (Tunnelling)", "**AWS** Cloud Architect".

By zooming in on these two categories, you can extract "Skills" directly from the Job Titles using Python Regex, effectively bypassing your data limitation.

### **New Project Proposal: "The Tech & Engineering Talent Radar"**

**The Elevator Pitch:**

"While other groups are looking at the generic job market, Group 6 provides deep competitive intelligence on Singapore's most critical sectors: Tech and Engineering. We don't just tell you *who* is hiring; we tell you *what specific technologies* (Python vs. Java, Tunnelling vs. Structural) are driving the economy."

#### **1. The Business Problem**

* **For CTOs & Engineering Directors:** "I need to hire a team. Is it cheaper to hire a Java team or a Python team in Singapore right now? Is the talent supply drying up?"
* **For Tech Recruiters:** "What is the salary premium for 'DevOps' keywords compared to standard 'SysAdmin' titles?"

#### **2. The Solution (Your Dashboard Views)**

Since you are focusing on 26% of the data (~270k rows), your dashboard will be faster and deeper.

* **View 1: The "Language Wars" (IT Focus)**
  + *Chart:* Line chart of Job Postings over time for specific keywords extracted from titles: Python, Java, React, .NET.
  + *Business Insight:* "Demand for 'React' has overtaken 'Angular' in Q3, signaling a market shift."
* **View 2: The "Salary Arbitrage" Map**
  + *Chart:* Box plot of Salary Distributions by Tech Stack.
  + *Business Insight:* "Python Developers command a 15% salary premium over PHP Developers."
* **View 3: The "Engineering Discipline" Split**
  + *Chart:* Breakdown of Engineering by sub-discipline (Civil, Mechanical, Electrical, Chemical) based on title keywords.
  + *Business Insight:* "Civil Engineering hiring is spiking (likely due to new MRT lines), while Mechanical is flat."

### **Technical Implementation: The "Title-to-Skill" Extractor**

You don't need NLP or AI. You need a robust **Keyword Dictionary** and a simple Python function to tag your rows.

Here is the Python logic to prove this is feasible:

Python

import pandas as pd  
import re  
  
# Sample Data (Simulating your IT/Eng subset)  
data = {  
 'job\_title': [  
 'Senior Python Developer',  
 'Junior Java Engineer',  
 'Civil Engineer (Structural)',  
 'DevOps Engineer (AWS)',  
 'Mechanical Eng',  
 'Project Manager' # Noise to filter out  
 ],  
 'salary': [8000, 4500, 5500, 9000, 5000, 7000]  
}  
df = pd.DataFrame(data)  
  
# 1. Define your "High Signal" Dictionaries  
tech\_keywords = {  
 'Python': r'\bpython\b',  
 'Java': r'\bjava\b',  
 'Cloud/AWS': r'\b(cloud|aws|azure)\b',  
 'React/Frontend': r'\b(react|angular|vue|frontend)\b',  
 'Data': r'\b(data|ai|machine learning)\b'  
}  
  
eng\_keywords = {  
 'Civil': r'\b(civil|structural|tunnel|bridge)\b',  
 'Mechanical': r'\b(mechanical|hvac)\b',  
 'Electrical': r'\b(electrical|power)\b'  
}  
  
# 2. Function to Extract Tags from Title  
def extract\_skill(title, keyword\_dict):  
 title\_lower = str(title).lower()  
 found\_skills = []  
 for skill, pattern in keyword\_dict.items():  
 if re.search(pattern, title\_lower):  
 found\_skills.append(skill)  
 return found\_skills if found\_skills else ['General/Other']  
  
# 3. Apply the Logic  
# Note: You would apply this ONLY to the filtered IT & Engineering dataframes  
df['Tech\_Stack'] = df['job\_title'].apply(lambda x: extract\_skill(x, tech\_keywords))  
  
# Explode the list so one job can count for multiple skills (e.g., "Python AWS Dev")  
df\_exploded = df.explode('Tech\_Stack')  
  
# 4. Result for Dashboard  
print(df\_exploded.groupby('Tech\_Stack')['salary'].mean())

### **Why this beats Group 9 (The "Skills Gap" Team)**

Group 9 is trying to do a "Skills Gap Analysis" for *everyone*.

* **Their weakness:** Without job descriptions, their "skills" analysis will be very shallow for generic roles.
* **Your strength:** By narrowing to IT/Eng, your "Title-as-Skill" proxy is extremely accurate. "Java Developer" *means* they need Java skills. You don't need the description to know that.

### **Strategic Recommendation for Next Steps**

1. **Filter your dataset immediately:** Create a new DataFrame df\_tech\_eng containing only "Information Technology" and "Engineering".
2. **Scan your titles:** Do a quick df\_tech\_eng['job\_title'].value\_counts().head(50) to see the most common words. This will help you build your tech\_keywords dictionary.
3. **Build the "Stack vs Salary" Chart:** This is your "Killer Feature." If you can show that "Cybersecurity" titles pay $2k more on average than "Software Engineering" titles, you have a solid business product.

Does this "Tech & Engineering" focus feel like a stronger direction for your team?