

AI MOCK INTERVIEWER FOR DATA ANALYTICS/DATA SCIENCE

TEAM 3:

CHARISHMA TAMARANA

KANCHAN ASHOK NAIK

PRANAV REVEENDRAN

PRIYA GOVINDARAJULU

VAISHNAVI NANDURI

SOUMYA CHALLURU SREENIVAS



AGENDA

- Project Background and Executive Summary
- Project Requirements
- Team Organization
- Technology and Solution Survey
- Literature Survey of Existing Research
- Project Development Methodology
- Project Organization Plan
- Project Resource Requirements and Plan
- Project Schedule
- Data Process
- Data Collection
- Data Pre-Processing
- Data Transformation
- Data Preparation
- Data Statistics
- Machine Learning Models
- Machine Learning Models Prototyping
- Results
- Project Demo Pipeline Demonstration

Project Background and Executive Summary

Background

- Interview prep tools for data roles often lack adaptability and realism. Our AI Mock Interviewer addresses this by simulating dynamic, role-specific interviews that evaluate both technical and soft skills.

Executive Summary

- The system supports four roles: **Data Scientist**, **Data Analyst**, **ML Engineer**, and **Data Engineer**. It uses **Gemini NER** to extract metadata from user inputs and retrieves relevant content from **ChromaDB** via **RAG**.
- Interview questions are generated using fine-tuned **LLaMA 3**, adapting in real time. Deployed via a **cloud ETL pipeline using Amazon S3**, the system is scalable and secure.
- **Upcoming features:** **Voice input with Whisper** and **resume-based customization**.

Project Requirements:

Overview

Data Science Interview

AI simulates real interviews for Data Science and Analytics roles.

Personalized Questions

Dynamic, adaptive, and personalized questions enhance the experience.

Multi-Modal Support

Platform supports both text and voice-based response interactions.

Real-Time Feedback

AI provides feedback on correctness, confidence, and clarity of answers.



Functional Requirements

User Authentication – Secure login & role-based access.

Dynamic Question Generation – AI tailors questions based on candidate background.

Adaptive Interviewing – Adjusts question difficulty based on performance.

Speech-to-Text & Text-to-Speech – Enables seamless voice interaction.

Response Recording & Review – Stores answers for future analysis.

Performance Dashboard – Tracks progress, feedback, and weak areas.

AI-Powered Intelligence

AI-Based Answer Evaluation – NLP models assess correctness & structure.

Sentiment & Emotion Analysis – Detects fluency, confidence, and hesitation.

Bias Mitigation – Ensures fair evaluations across demographics.

Adaptive Learning – AI refines question difficulty over time.

Multi-Turn AI Conversations – Generates follow-up questions for deeper evaluation.

Data Requirements

Training Data – Uses Kaggle, Stack Overflow, and real-world interviews.

Speech & Response Storage – Secure cloud storage with encryption.

Vector Databases (FAISS, ChromaDB) – Efficient question retrieval.

Preprocessing & Data Cleaning – Normalization, denoising, and tokenization.

Performance Tracking – Logs candidate progress for AI improvements.

Team Organization & Function Roles

Function	Role	Team Member
Project Planning and Management	Project Planner	KANCHAN ASHOK NAIK
Data Collection and Processing Management	Data Manager	CHARISHMA TAMARANA
Frontend UI / Backend Integration	UI Manager	PRIYA GOVINDARAJULU
Storage /Pipeline Monitoring	Pipeline Manager	PRANAV REVEENDRAN
Integration and Testing	QA and testing Manager	VAISHNAVI NANDURI
Research and Review	Knowledge Base Architect	SOUMYA CHALLURU SREENIVAS
Data Collection/Literature Survey research/ Model development/ Coding/ RAG /Pipeline Development /UI Creation	Generic Tasks	EVERYONE (All Team Members)

Technology and Solution Survey

- **AI-Driven Interviewers** – LLMs (GPT-4o, Claude 2) adapt questions dynamically using reinforcement prompting.
- **Multi-Agent Systems** – Role, Environment, and Memory agents improve interview flow and response evaluation.
- **Skill Extraction & Matching** – Uses **KVMN** and **Masked Self-Attention** for resume-job relevance.
- **Evaluation & Optimization** – Code accuracy (Pass@K), **Emotion & Confidence Analysis**, and **Audio-to-Text (Whisper Model, WER)**.
- **Fine-Tuning** – LoRA, 4-bit LLaMA3, Planning-based QG (BART, T5) optimize performance.

Literature Survey: AI-Powered Mock Interview Systems

Agrawal et al. (2020)

Leveraging Multimodal Behavioral Analytics for Automated Job Interview Performance Assessment

Chou et al. (2022)

An AI Mock-Interview Platform for Interview Performance Analysis

Mandal et al. (2023)

AI-Based Mock Interview Evaluator: An Emotion and Confidence Classifier Model

Rao et al. (2020)

Automatic Follow-up Question Generation for Asynchronous Interviews

Li et al. (2023)

EZInterviewer: To Improve Job Interview Performance with Mock Interview Generator

Chavan et al. (2024)

AI-Based Mock Interview System

Patil (2024)

Elevating Performance Through AI-Driven Mock Interviews

Ghodichor et al. (2024)

AI-Powered Mock Interview Evaluator

Nag et al. (2024)

AI-Driven Mock Interview: A New Era In Candidate Preparation

Sun et al. (2024)

Facilitating Multi-Role Collaboration of LLMs for Online Job Seeking and Recruiting

Project Development Methodology

- Followed a **hybrid CRISP-DM + Agile** approach for structured yet iterative development.
- Targeted four roles: **Data Scientist, Data Analyst, ML Engineer, Data Engineer**.
- Fine-tuned **LLaMA 3** for role-specific, adaptive question generation.
- Used **Gemini NER** to extract metadata like company, role, and interview round from user inputs.
- Implemented **Retrieval-Augmented Generation (RAG)** using **ChromaDB** to ground questions in contextually relevant data.
- Built a **cloud-based ETL pipeline** using **Amazon S3** for scalable data processing and transformation.
- Evaluation involved **perplexity, semantic scoring (e.g., BERTScore)**, and **human expert review**.
- Planned future integration of **OpenAI Whisper** to support **voice-based interaction**.

Project Plan

Phases & Deliverables

Phase	Duration	Key Deliverables
Phase 1: Planning	Week 1	Model selection, dataset planning, UI wireframes
Phase 2: Data Preparation	Week 2-3	Interview QA pairs, metadata - Company, role and round, prompt templates, formatting
Phase 3: RAG and Fine-Tuning	Weeks 4-5	Fine-tuned versions of Mistral and Llama for Data Science role (LoRA)
Phase 4: Integration	Week 6	Unified inference backend, frontend connection
Phase 5: Evaluation	Week 7	Collect user responses, analyze outputs per model

Project Resource Requirements and Plan

Resource Type	Function	Resource	Usage Time Duration	Estimated Cost (USD)
Hardware	Model Fine-tuning	GCP compute engine(8 vCPUs, 32GB RAM, 1TB SSD)	3 months (September 2025 to December 2025)	\$925.2 (\$309.40 per month)
Hardware	Model Fine-tuning	Google Colab Pro(GPU-enabled, NVIDIA T4/V100 GPUs)	3 months (September 2025 to December 2025)	\$29.97 (\$9.99 per month)
Hardware	For connecting to cloud services and development.	Personal System(32GB RAM, 1TB SSD, NVIDIA RTX 3090 GPU)	Used as required	Free
Hardware	Cloud storage	GCP cloud storage standard and archival(600GB) / AWS storage	10 months(March 2025 to December 2025)	\$110.8 (\$11.08 per month)
Tools	version control	GitHub	10 months(March 2025 to December 2025)	Free
Tools	Local project development tool	Jupyter Lab	10 months(March 2025 to December 2025)	Free
Tools	Documentation and Project Management	Microsoft Office suite, Monday.com	10 months(March 2025 to December 2025)	Free

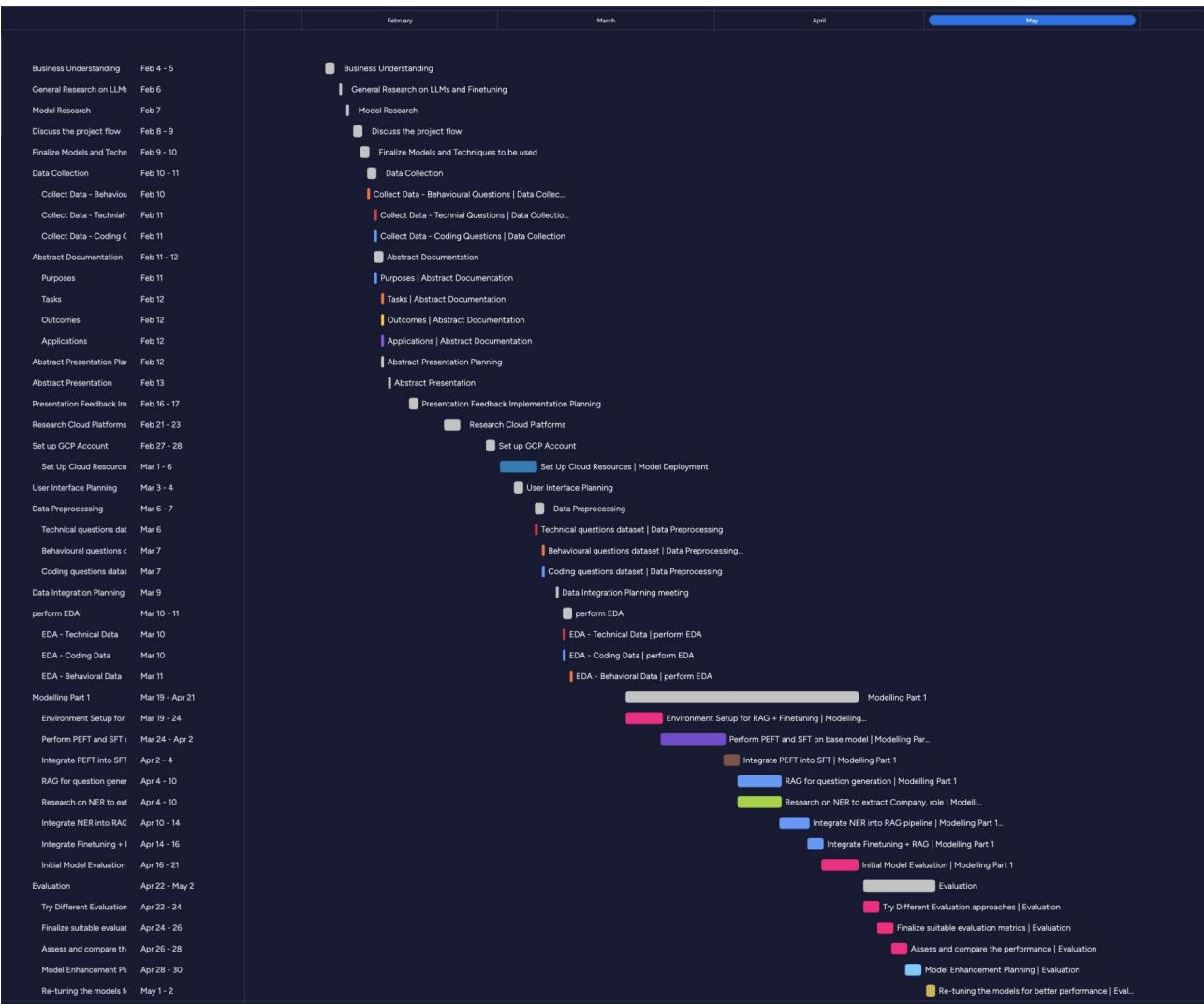
Project Resource Requirements and Plan

Resource Type	Function	Resource	Usage Time Duration	Estimated Cost (USD)
Tools	Team meetings	Zoom	10 months(March 2025 to December 2025)	Free
Software	Speech-to-text processing	Open AI Whisper	3 months (September 2025 to December 2025)	\$300 (\$100 per month)
Software	LLM for fine-tuning and Project development	LLaMA-2, Mistral, PyTorch, TensorFlow, LoRA/QLoRA	4 months (August 2025 to December 2025)	Free
Software	Vector Database/ Front end UI/backend API	Pinecone / React / node.js / Fast API / HTML/ CSS	3 months (September 2025 to December 2025)	Free(Starter Plan)
Software	Cloud Management	Google Cloud SDK	10 months(March 2025 to December 2025)	Free
Software	Cloud database	Google Cloud SQL/Snowflake	3 months (September 2025 to December 2025)	\$346.86 (\$115.62 per month)
Software	Project Development and audio pre-processing.	Python, Pandas, NumPy, Scikit-Learn, PyTorch, Tensorflow, Librosa, Airflow	10 months(March 2025 to December 2025)	Free
Total Cost:				\$1712.83

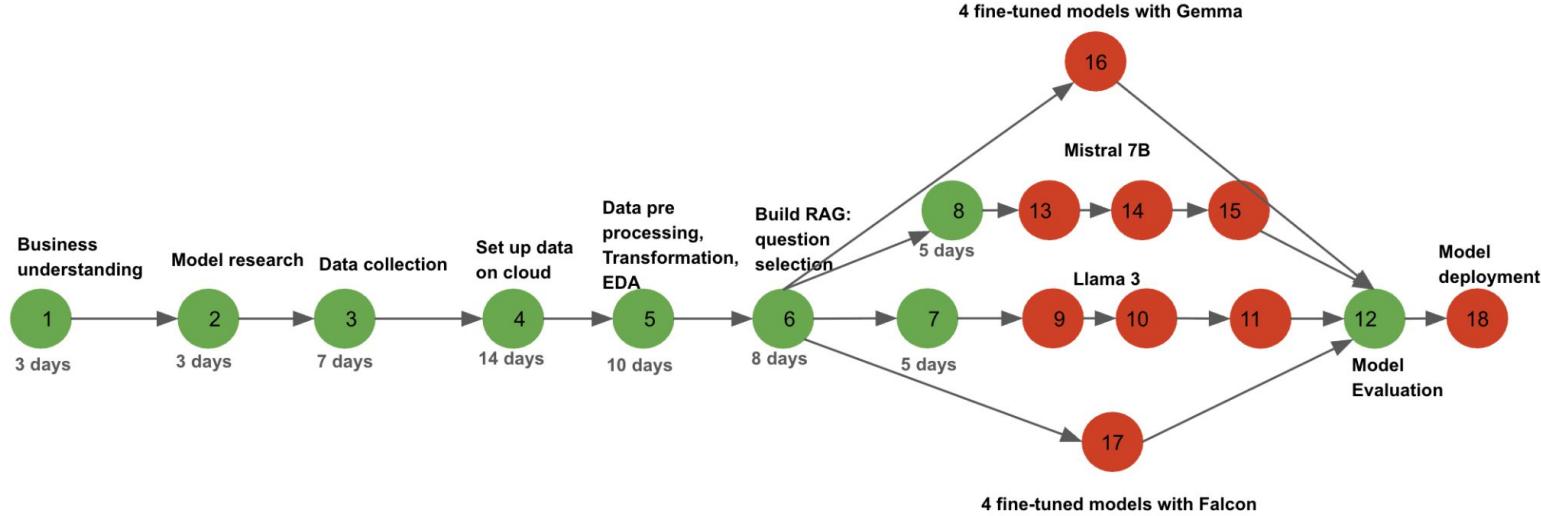
Project Schedule

Model Deployment		Next Phase	May 1
Subitem	Owner	Status	Date
Set Up Cloud Resources	Pranav	Done	Mar 5
Deploy models on cloud	Everyone	Next Phase	Sep 4
Final Documentation	Everyone	Next Phase	Dec 2

- Tasks
- Charishma, Pranav
 - Everyone
 - Kanchan
 - Kanchan, Vaishnavi
 - Pranav
 - Priya, Kanchan
 - Priya, Pranav
 - Soumya, Charishma
 - Soumya, Charishma, Pranav
 - Vaishnavi
 - Vaishnavi, Kanchan
 - Vaishnavi, Kanchan, Priya



Project Flow



Node 7 - Fine-tuned Llama for Data Science role
Node 8 - Fine-tuned Mistral for Data Science role

Node 9 - Fine-tuned Llama for Data Engineer role
Node 10 - Fine-tuned Llama for ML Engineer role
Node 11 - Fine-tuned Llama for Data Analyst role

Node 13- Fine-tuned Mistral for Data Engineer role
Node 14 - Fine-tuned Mistral for ML Engineer role
Node 15 - Fine-tuned Mistral for Data Analyst role

Data Process

1. Initial Exploration of Public Datasets

- publicly available datasets from platforms such as Kaggle, Hugging Face, and GitHub
- Few were either outdated or lacked critical context such as the company and role level
- limited data – for company and role specific

2. Web Scraping for High-Quality and Real-World Data

- Glassdoor, GeeksForGeeks , LinkedIn, Reddit, LeetCode and StackOverflow
- BeautifulSoup/Selenium python scripts to scrape data
- Web Scraper.io and Instant Data Scraper

3. Integration and Categorization of questions to form a dataset

- Brought up all questions into S3 Bucket
- Assigned Each question- Company, role, Category and Type
- The main categories we defined were Technical, Programming, and Behavioral

4. Planning and Finalizing Dataset Structure

- We finalized a consistent structure for each data entry to ensure uniformity across all sources
- follow a uniform schema, with consistent column names and data types
- Tags Column: Company, role, category and type in one single column as list

Data Collection



Text Data :

Data Sourced from sources such as Kaggle, GitHub, Glassdoor, Reddit, Hugging Face, Stack Overflow, and blogs.

The raw data includes **157,741** Q&A pairs for behavioral, technical, and programming interviews on domains such as Data Science, Data Engineering and Data Analyst.

Metadata: Domain, Difficulty Level, Interview Type, Company, Role



Voice Data:

Sample voice data taken from **MIT Interview Dataset** (Naim et al., 2018) for preprocessing and training Whisper AI.

The sample voice dataset contains **138** audio files from mock interviews with 69 students (.wav format)

Metadata: audio files, Ground Truth Label with text data for the interview audio files

Data Preprocessing

Data Cleaning

The duplicate rows with repeated questions are verified and deleted if present.
The null values in metadata columns such as domain and interview type were filled.
The unwanted columns, such as empty or extra columns, were identified and removed.

Text Preprocessing

The contractions are expanded, from “don’t” to “do not”.
The unwanted symbols and the special characters were removed.
Tokenization is applied to the text data to split the data into tokens.
Lemmatization is used to reduce the root words form (e.g., “running” to “run”)

Audio Preprocessing

The noise in voice data is denoised using Librosa.
The volume of the audio file is normalized to standard level (-20 dBFS).
Files are resized to keep it under 20MB for compatibility with OpenAI Whisper.

Data Transformation

Data was transformed into structured formats for **technical**, **programming**, **behavioral**, and (planned) **voice** modules. ETL pipelines converted CSVs to enriched JSON with metadata, embeddings, and complexity tags.

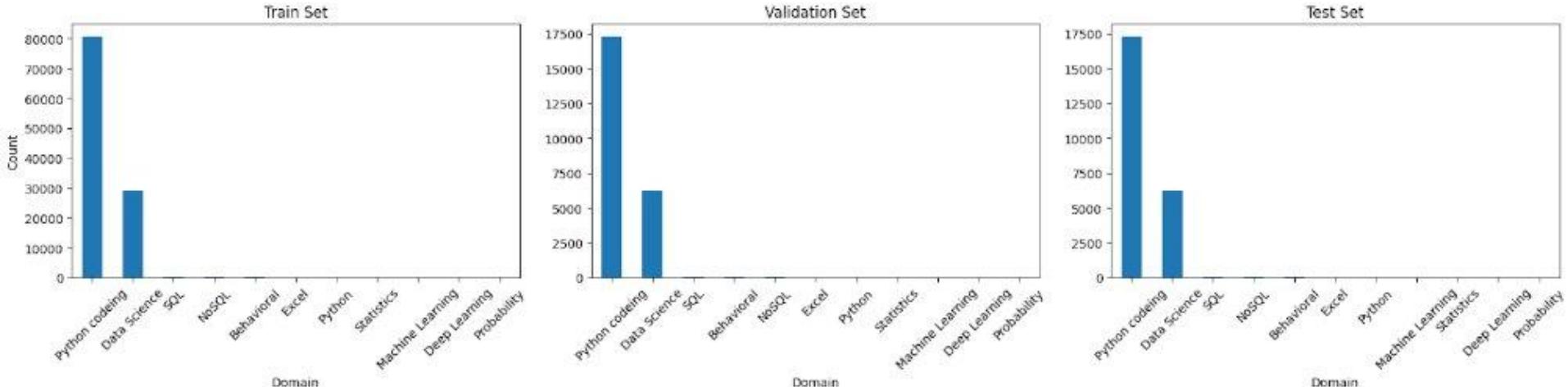
Missing or incomplete entries were removed to ensure data quality.

- **Technical:** Labeled via cosine similarity; added embeddings and word counts.
- **Programming:** Code cleaned (Black), tagged, and stored in normalized tables.
- **Behavioral:** Embedded, sentiment-tagged, and categorized (e.g., leadership) for soft skill learning.
- **Voice (Planned):** Will use **Whisper** for transcription and **Parselmouth** for vocal features.

Data Preparation

- Finalized datasets for all modules: Technical, Programming, Behavioral, and Voice
- Dataset size: 157,000+ entries
- Applied standard split: 70% Train / 15% Validation / 15% Test
- Used stratified sampling to balance key attributes like Domain, Company, Level
- Employed Scikit-learn's `train_test_split()` with `stratify` for even distribution

Domain Distribution in Train, Validation, and Test Sets



```
<ipython-input-15-728910840661>:6: DtypeWarning: Columns (6) have mixed types. Specify dtype option on import or set low_memory=False.
  df = pd.read_csv(file_path)
→ Remaining rows after drop: (157717, 14)
Train Set: (10481, 14)
Validation Set: (23658, 14)
Test Set: (23658, 14)

Sample from Training Set:
```

	Question	Answer	Domain
113895	Create a function in Python to extract all num...	def extract_numbers(s):\n return [int(e) ...	Python codeing
37951	Extract list of attributes from list of object...	atrrs = [o.attr for o in objs] was the right c...	Python codeing
3838	Write a python program to generate the binary ...	def decimal_to_binary(num):\n if num > 1:\n ...	Python codeing
153756	Analysis with categorical variable	You could use the following different methods...	Data Science
32369	While Patrick was gone shopping, Spongebob dec...	["def calc(f, b):\n nta = [0 for i in range(100...]	Python codeing

```
Sample from Validation Set:
```

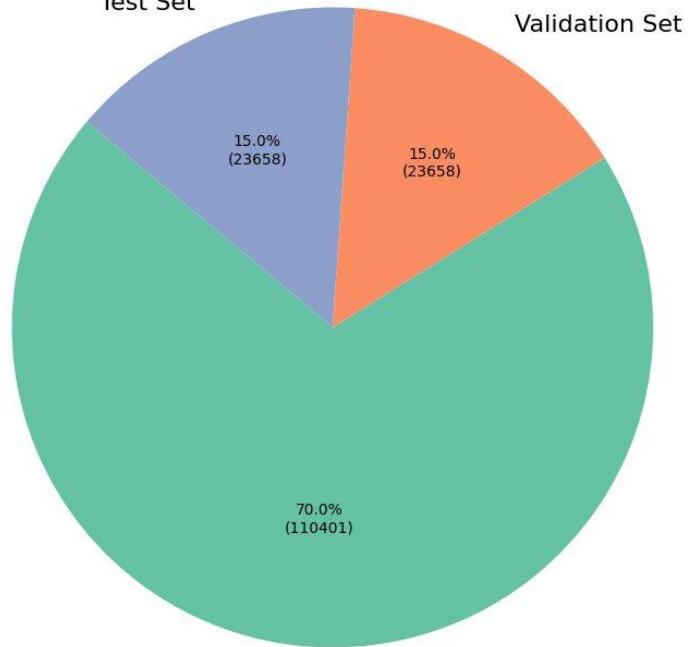
	Question	Answer	Domain
6316	Optimize a function in Python which takes a li...	def sum_list_elements(lst):\n sum = 0\n ...	Python codeing
128831	Tensorflow Calculate error for a single neuron	Easy, you have already done it.\nmse = tf.redu...	Data Science
80802	In India IPL is on full swing now. Fans of cri...	"""\n# Read input from stdin and provide inp...	Python codeing
18752	oolimty has an array \$a\$ of length \$n\$ which h...	["import sys\n\ndef solve():\n inp = sys.stdin...	Python codeing
960	Write a function to flatten a given nested lis...	def flatten_list(list1):\n result_list = ...	Python codeing

```
Sample from Test Set:
```

	Question	Answer	Domain
40715	Improving Performance of Django ForeignKey Fie...	Add raw_id_fields to your model to only show t...	Python codeing
963	Write a function in Python that encodes a stri...	def encodeString (text):\n encoded_string = ...	Python codeing
17512	Since you are the best Wraith King, Nizhniy Ma...	["str = input().split()\nn = int(str[0])\nlen ...	Python codeing
76576	Read problem statements in [Hindi], [Bengali], ...	["for _ in range(int(input())):\n n = int(inp...]	Python codeing
92504	AttributeError: StringIO instance has no attri...	The fileno() method is not implemented in Stri...	Python codeing

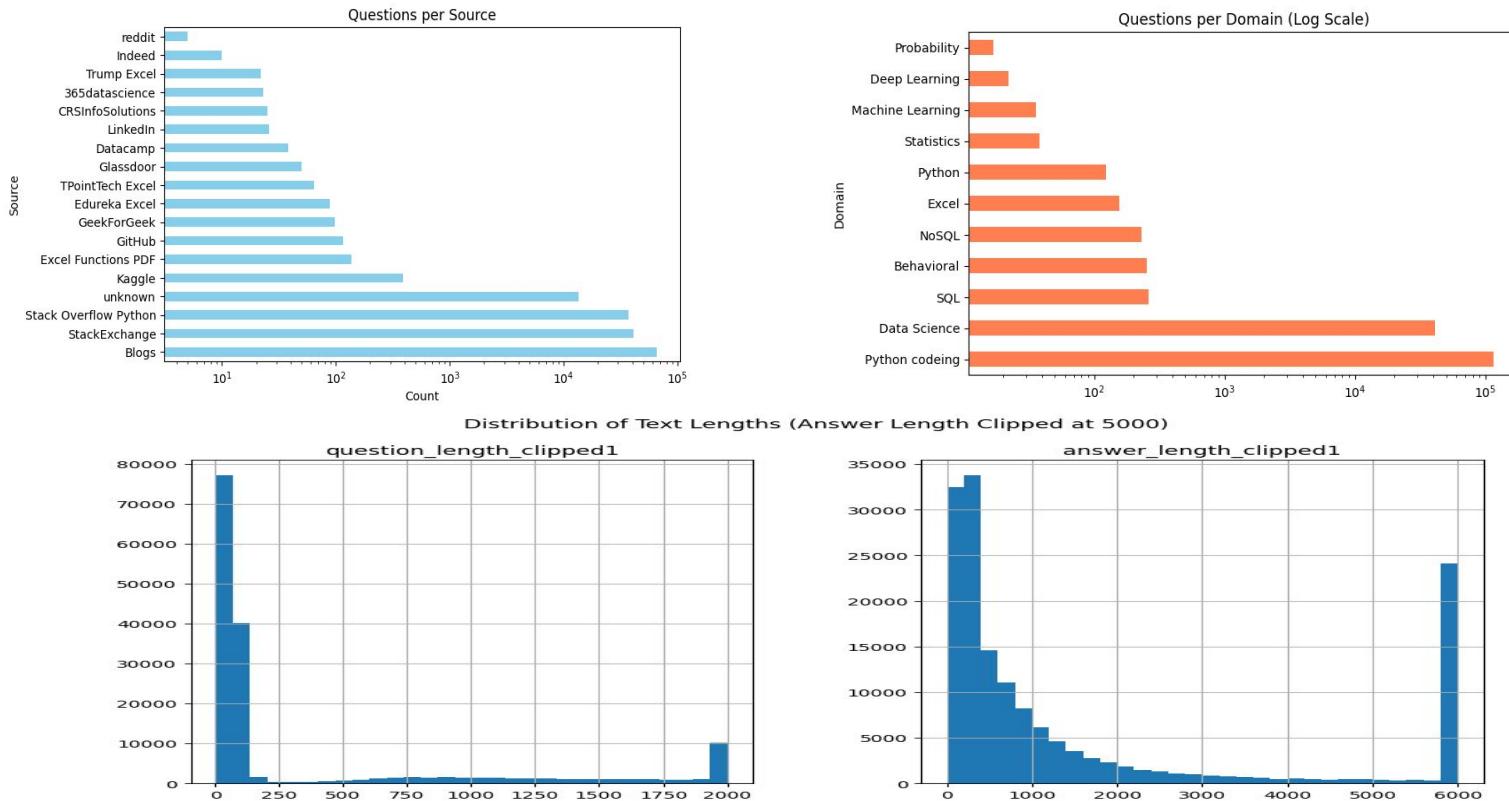
Dataset Split: Training vs Validation vs Test

Test Set



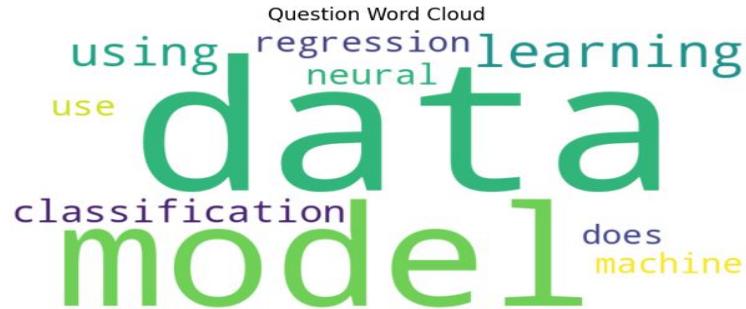
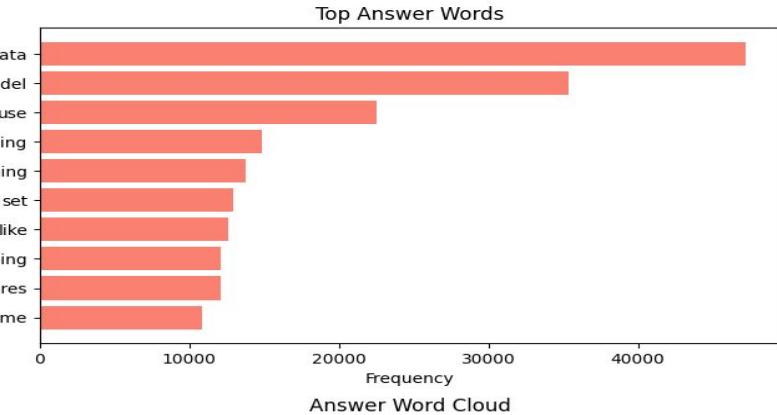
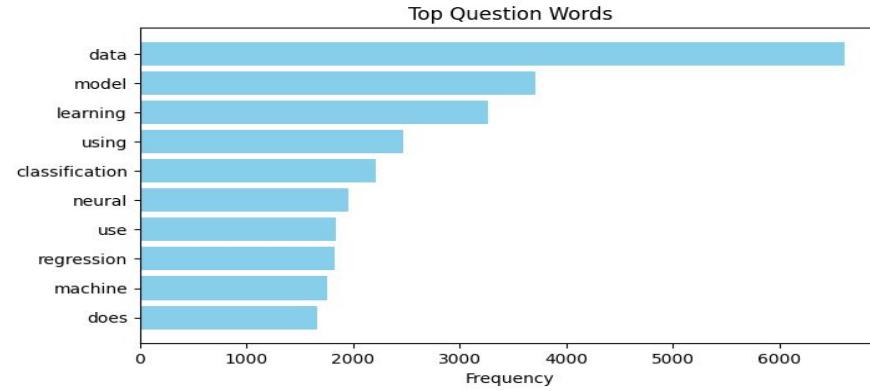
Training Set

Data Statistics



Data Statistics

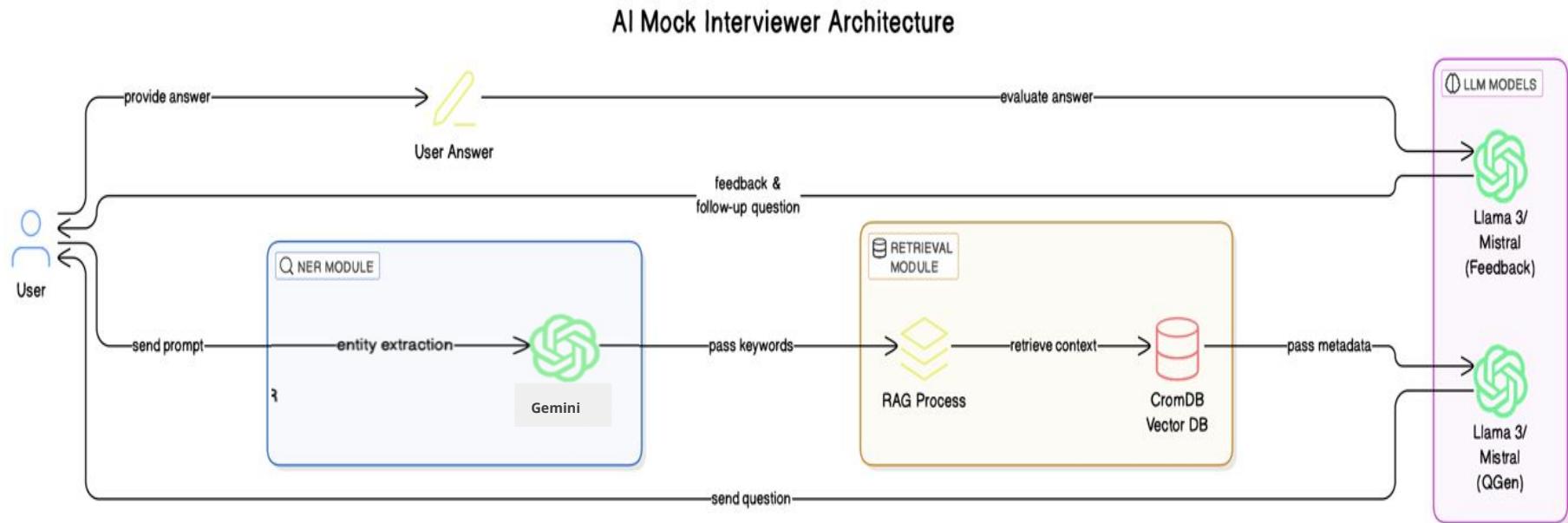
Top Words Summary for Domain: Data Science



Fine-tune models

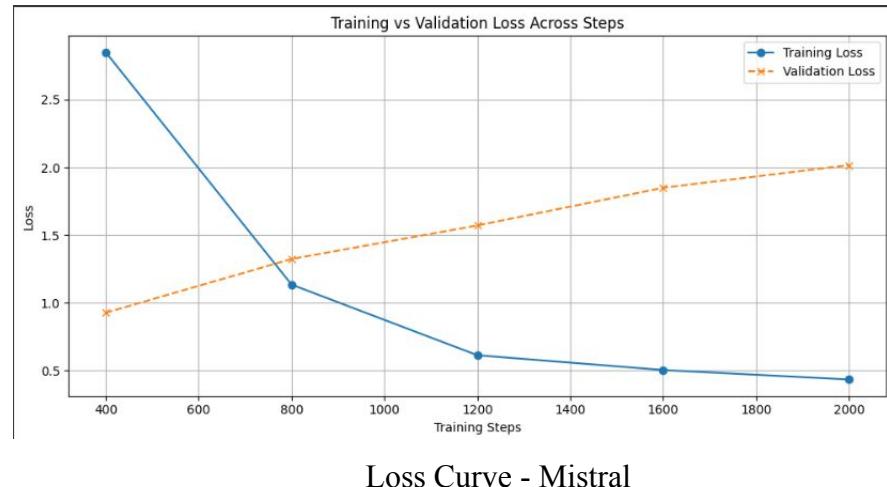
- Data Analytics Interview Model
- Data Scientist Interview Model
- Data Engineer Interview Model
- Machine Learning Engineer Interview Model

Architecture Diagram



LLM Models Prototyping - Llama3 & Mistral

- PEFT fine Tune 4 bit quantized model
- Trained on generating constructive feedback on answers and follow-up question
- GPT 4 for bootstrapping feedback



Comparative Performance of Fine tune models

- LLama 3 (Meta-Llama-3-8B-Instruct) - Perplexity: 6.37
- Mistral (Mistral-7B-Instruct-v0.2) - Perplexity: 8.74
- Gemma (For future)
- Falcon (For future)

Company, Round & Role Specific Data

	A	B	C	D	E	F	G	H	I	J
1	Company	Role	Tags	Round Number	Round Name	Topic Questions				
2	Meta	Machine Learning Engineer	Behavioral,Machinelearning,Programminglanguages,Frameworks	Round 1	HR Interview	Why do you want to join Meta?				
3	Meta	Machine Learning Engineer	Behavioral,Machinelearning,Programminglanguages,Frameworks	Round 1	HR Interview	Why do you think you will be a good fit for the role?				
4	Meta	Machine Learning Engineer	Behavioral,Machinelearning,Programminglanguages,Frameworks	Round 1	HR Interview	What responsibilities do you expect to have from your job at Meta?				
5	Meta	Machine Learning Engineer	Behavioral,Machinelearning,Programminglanguages,Frameworks	Round 1	HR Interview	What are your areas of expertise in machine learning, and how have you developed those skills?				
6	Meta	Machine Learning Engineer	Behavioral,Machinelearning,Programminglanguages,Frameworks	Round 1	HR Interview	What programming languages are you comfortable with, and how have you used them in your previous				
7	Meta	Machine Learning Engineer	Behavioral,Machinelearning,Programminglanguages,Frameworks	Round 1	HR Interview	What machine learning frameworks have you used, and how did you find them helpful in your projects?				
8	Meta	Machine Learning Engineer	Behavioral,Machinelearning,Programminglanguages,Frameworks	Round 1	HR Interview	Have you worked with large datasets?				
9	Meta	Machine Learning Engineer	Behavioral,Machinelearning,Programminglanguages,Frameworks	Round 1	HR Interview	If so, how did you handle data preprocessing and ensure data quality?				
10	Meta	Machine Learning Engineer	Behavioral,Machinelearning,Programminglanguages,Frameworks	Round 1	HR Interview	Have you implemented any machine learning algorithms from scratch?				
11	Meta	Machine Learning Engineer	Behavioral,Machinelearning,Programminglanguages,Frameworks	Round 1	HR Interview	If so, can you provide an example and explain your thought process?				
12	Meta	Machine Learning Engineer	Coding,Datastructures,Algorithms,Bigo,Python	Round 2	Technical Screen	Can you walk me through how you would approach solving a coding problem?				
13	Meta	Machine Learning Engineer	Coding,Datastructures,Algorithms,Bigo,Python	Round 2	Technical Screen	How would you ensure that your code is optimized and free of logic flaws?				
14	Meta	Machine Learning Engineer	Coding,Datastructures,Algorithms,Bigo,Python	Round 2	Technical Screen	Can you provide an example of a challenging problem you solved with a creative solution?				
15	Meta	Machine Learning Engineer	Coding,Datastructures,Algorithms,Bigo,Python	Round 2	Technical Screen	What are your strengths when it comes to data structures and algorithms?				
16	Meta	Machine Learning Engineer	Coding,Datastructures,Algorithms,Bigo,Python	Round 2	Technical Screen	How do you handle ambiguity in problem-solving?				
17	Meta	Machine Learning Engineer	Coding,Datastructures,Algorithms,Bigo,Python	Round 2	Technical Screen	Can you explain the concept of Big O notation and its significance in coding?				
18	Meta	Machine Learning Engineer	Coding,Leetcode,Datastructures,Debugging	Round 3	Onsite Interview - Coding	Design a data structure that supports insert, delete, search and get random in constant time., Find the				
19	Meta	Machine Learning Engineer	Coding,Leetcode,Datastructures,Debugging	Round 3	Onsite Interview - Coding	Implement a function to construct a binary tree., Write a code for the dot product of two sparse vectors				
20	Meta	Machine Learning Engineer	Coding,Leetcode,Datastructures,Debugging	Round 3	Onsite Interview - Coding	Given an infinite chessboard, find the shortest distance for a knight to move from position A to position				
21	Meta	Machine Learning Engineer	Systemdesign,Scalability,Architecture,Communication	Round 3	Onsite Interview - System Design	Design a scheduler for distributed cron jobs., Build a recommendation based engine., Design ML system				
22	Meta	Machine Learning Engineer	Ml,systemdesign,Modeling,Deployment,Featureengineering	Round 3	Onsite Interview - ML Design	What are the roles of batch normalization and dropout in training deep neural networks?				

Creating Embeddings and Storing in ChromDB

```
# Step 2: Load Excel
import pandas as pd
filename = list(uploaded.keys())[0]
df = pd.read_excel(filename)

# Step 3: Clean whitespace in key columns
for col in ['Company', 'Role', 'Round Name', 'Topic Questions']:
    df[col] = df[col].astype(str).str.strip()

# Step 4: Create combined text for embedding
df['combined_text'] = df.apply(
    lambda row: f"{row['Company']} - {row['Role']} - {row['Round Name']} - {row['Topic Questions']}",
    axis=1
)

# Step 5: Install and load embedding model
!pip install -q sentence-transformers
from sentence_transformers import SentenceTransformer
import json

model = SentenceTransformer('all-MiniLM-L6-v2')

# Step 6: Generate embeddings
df['embedding'] = df['combined_text'].apply(lambda x: model.encode(x).tolist())

# Step 7: Drop combined text
df.drop(columns=['combined_text'], inplace=True)
```

```
[ ] # Step 3: Initialize Chroma PersistentClient
persist_dir = "./chroma_db"
os.makedirs(persist_dir, exist_ok=True)

from chromadb import PersistentClient
client = PersistentClient(path=persist_dir)

# Step 4: Create or get a collection
collection = client.get_or_create_collection(name="interview_questions")

# Step 5: Load JSONL file
records = []
with open('processed_interview_with_embeddings.jsonl', 'r') as f:
    for line in f:
        record = json.loads(line)
        records.append(record)

# Step 6: Prepare and add data
texts = [r['Topic Questions'] for r in records]
embeddings = [r['embedding'] for r in records]
metadata = [{k: v for k, v in r.items() if k not in ['embedding', 'Topic Questions']} for r in records]

collection.add(
    documents=texts,
    embeddings=embeddings,
    metadata=metadata,
    ids=[f"q_{i}" for i in range(len(texts))]
)
```

Data Stored in ChromaDB

The screenshot shows a Jupyter Notebook interface with a sidebar on the left and a main workspace on the right.

Files sidebar:

- Analyze your files with code written by Gemini
- Upload
- File operations icons: Up, Download, Refresh, Copy, Cut, Paste, Delete
- File tree:
 - ..
 - chroma_db
 - 843e38b9-d878-4910-b3a7-29eb21...
 - data_level0.bin
 - header.bin
 - length.bin
 - link_lists.bin
 - chroma.sqlite3
 - sample_data
 - processed_interview_data-4.xlsx
 - processed_interview_with_embeddings...

Main workspace:

```
[6] # Step 7: Print collection size to confirm data is present
count = collection.count()
print(f"Documents in 'interview_questions' collection: {count}")

Documents in 'interview_questions' collection: 657

[7] # Step 8: Retrieve and print 2 sample entries from the collection
results = collection.get(
    ids=["q_0", "q_1"],
    include=["documents", "metadatas", "embeddings"]
)

for i in range(len(results['ids'])):
    print(f"--- Sample {i+1} ---")
    print(f"ID: {results['ids'][i]}")
    print(f"Question: {results['documents'][i]}")
    print(f"Metadata: {results['metadatas'][i]}")
    print()
```

Execution results:

```
--- Sample 1 ---
ID: q_0
Question: Why do you want to join Meta?
Metadata: {'Round Name': 'HR Interview', 'Role': 'Machine Learning Engineer', 'Round Number': 'Round 1', 'Tags': 'B'

--- Sample 2 ---
ID: q_1
Question: Why do you think you will be a good fit for the role?
Metadata: {'Company': 'Meta', 'Round Number': 'Round 1', 'Round Name': 'HR Interview', 'Role': 'Machine Learning En
```

Results

```
[ ] # Read and preview JSONL file in Colab
import json

with open('processed_interview_with_embeddings.jsonl', 'r') as f:
    lines = f.readlines()

# Parse first few JSON objects
for i, line in enumerate(lines[:5]):
    data = json.loads(line)
    print(f"\n--- Record {i+1} ---")
    print(json.dumps(data, indent=2))
```

```
→ --- Record 1 ---
{
  "Company": "Meta",
  "Role": "Machine Learning Engineer",
  "Tags": "Behavioral,Machinelearning,Programminglanguages,Frameworks",
  "Round Number": "Round 1",
  "Round Name": "HR Interview",
  "Topic Questions": "Why do you want to join Meta?",
  "embedding": [
    -0.028817638754844666,
    0.0369221568107605,
    0.01961103081703186,
    0.04577894136309624,
    0.15243588387966156,
    -0.045289553701877594,
    0.041193582117557526,
    0.012242757918446823,
```

RAG Embeddings

```
[ ] # 2. Metadata extractor using Gemini prompt
def extract_metadata_from_query():
    user_query = input("Enter your interview query: ")
    prompt = f"""
Extract the following information from the sentence below:

Sentence: "{user_query}"

Return a JSON object with:
- Company
- Role
- Round Number

Format:
{{{
    "Company": "...",
    "Role": "...",
    "Round Number": "..."
}}}
"""

    try:
        response = model.generate_content(prompt)
        response_json = response.text.strip()
        match = re.search(r'\n.*\n', response_json, re.DOTALL)
        if match:
            metadata = json.loads(match.group())
            print(" Extracted Metadata:", metadata)
            return metadata
        else:
            print("Could not find valid JSON in response.")
            return None
    except Exception as e:
        print(f"An error occurred: {e}")
        return None
```

Gemini for NER

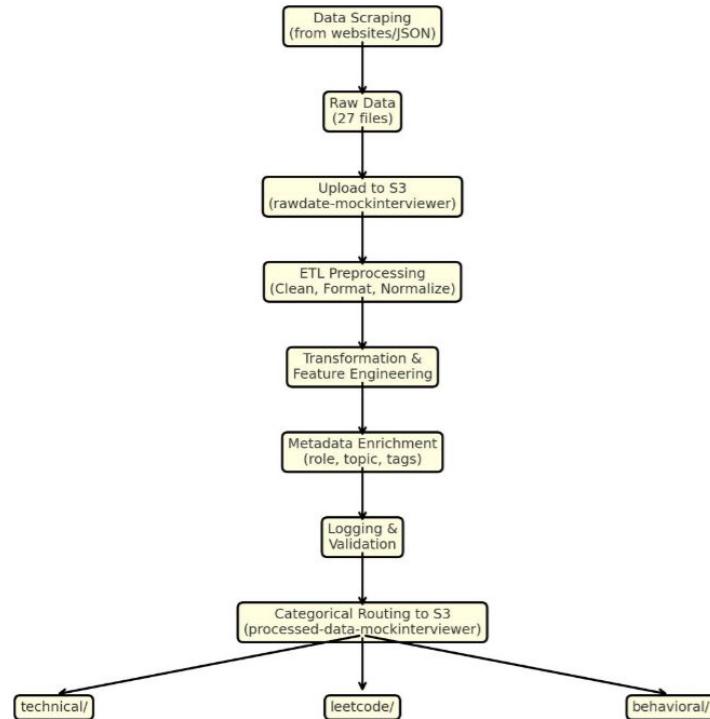
Results

```
▶ run_full_pipeline()  
→ Enter your interview query: Could you help me practice for Data Scientist Round 3 interview for Meta  
Extracted Metadata: {'Company': 'Meta', 'Role': 'Data Scientist', 'Round Number': '3'}  
🔍 Querying: interview questions for Data Scientist Round 3  
--- Match 1 ---  
Question: Which data sets would you use to answer specific product-related questions?  
Metadata: {'Round Name': 'Analytical Reasoning Interview', 'Role': 'Data Scientist', 'Tags': 'Behavioural,Statistics,ProductCase,ExperimentDesign,BiasDetection,DataIntegration,CodeOptimization,Communication,Metrics', 'Company': 'Meta', 'Round Number': '3'}  
--- Match 2 ---  
Question: Given a dataset on user engagement, how would you structure a solution to improve user interaction on our platform?  
Metadata: {'Tags': 'Behavioural,Statistics,ProductCase,Sql,Coding,EdgeCases,DataAdaptation,CodeOptimization,Communication,Metrics', 'Round Name': 'Analytical Reasoning Interview', 'Role': 'Data Scientist', 'Company': 'Meta', 'Round Number': '3'}  
--- Match 3 ---  
Question: How do you draw meaningful conclusions from a dataset?  
Metadata: {'Round Name': 'Analytical Reasoning Interview', 'Tags': 'Behavioural,Statistics,ProductCase,ExperimentDesign,BiasDetection,DataIntegration,CodeOptimization,Communication,Metrics', 'Company': 'Meta', 'Round Number': '3'}  
--- Match 4 ---  
Question: How do you integrate information from various sources into a cohesive insight?  
Metadata: {'Company': 'Meta', 'Round Name': 'Analytical Reasoning Interview', 'Role': 'Data Scientist', 'Tags': 'Behavioural,Statistics,ProductCase,Sql,Coding,EdgeCases,DataAdaptation,CodeOptimization,Communication,Metrics', 'Round Number': '3'}  
--- Match 5 ---  
Question: Code a solution to analyze customer behavior data and identify patterns that could inform marketing strategies., Consider a scenario where we want to predict future purchases based on past buying habits.  
Metadata: {'Tags': 'Behavioural,Statistics,ProductCase,Sql,Coding,EdgeCases,DataAdaptation,CodeOptimization,Communication,Metrics', 'Role': 'Data Scientist', 'Company': 'Meta', 'Round Number': '3'}
```



Q and A?

Data pipeline Demo - Entire Pipeline Diagram



Data pipeline Demo:

The screenshot shows a web-based data pipeline monitoring interface with a dark theme. On the left is a sidebar with icons for Home, Dags (selected), Assets, Browse, Admin, Docs, and User. The main area has tabs for Dags, Runs, and Task Instances, with Dags selected. A search bar at the top allows searching for Dags and provides an Advanced Search option. Below the search bar are filters for Failed, Running (green), Success (green checkmark), and All runs. There is also a 'Filter by tag' dropdown and a 'Reset 1 filter' button. A sorting dropdown is set to 'Sort by Latest Run Start Date...'. The main content area displays three data aggregation dags:

- data_aggregation_dag** (aggregation, mockinterviewer, output)
 - Schedule: 0 0 * * *
 - Latest Run: 2025-05-12, 03:41:41 (Success)
 - Next Run: 2025-05-12, 17:00:00
- feature_engineering_dag** (engineering, mockinterviewer, feature)
 - Schedule: 0 0 * * *
 - Latest Run: 2025-05-12, 03:41:40 (Success)
 - Next Run: 2025-05-12, 17:00:00
- etl_preprocessing_dag** (etl, mockinterviewer, preprocessing)
 - Schedule: 0 0 * * *
 - Latest Run: 2025-05-12, 03:41:38 (Success)
 - Next Run: 2025-05-12, 17:00:00

Data Pipeline Demo:

localhost:8080/dags/data_aggregation_dag

Dag data_aggregation_dag

Home Dags Assets Browse Admin Docs User

Options

7s 3s 0s

start load_enriched_feat... validate_records export_to_jsonl upload_to_vectorst... notify_pipeline_ready end

Schedule Latest Run Next Run Owner Tags

0 0 * * * 2025-05-12, 03:41:41 2025-05-12, 17:00:00 airflow aggregation, mockinterviewer, output

Latest Dag Version v2

Trigger Search Dags

Run Backfill Reparse Dag

Overview Runs Tasks Backfills Events Code Details

Last 24 hours 2025-05-11, 03:42:50 - 2025-05-12, 03:42:50

Last 3 Dag Runs

Run After	Duration (seconds)
2025-05-11, 05:00:00	6.75
2025-05-12, 03:40:03	0.64
2025-05-12, 03:41:41	

Data Pipeline Demo:

The screenshot shows the Airflow web interface for the `feature_engineering_dag`. The left sidebar has a dark theme with blue highlights for the `Dags` section. The main page displays the DAG configuration and its recent runs.

DAG Details:

- Dag ID:** feature_engineering_dag
- Schedule:** 0 0 * * *
- Latest Run:** 2025-05-12, 03:41:40 (Success)
- Next Run:** 2025-05-12, 17:00:00
- Owner:** airflow
- Tags:** engineering, mockinterviewer, feature

Latest Dag Version: v2

Overview Tab: Selected. Shows the last 24 hours from 2025-05-11, 03:43:05 to 2025-05-12, 03:43:05. A chart titled "Last 3 Dag Runs" shows the duration of three runs: 2025-05-11, 05:00:00 (Duration: 5.73 seconds), 2025-05-12, 03:39:56 (Duration: 0.52 seconds), and 2025-05-12, 03:41:40 (Duration: 5.73 seconds).

Runs Tab: Shows the latest run details.

Tasks Tab: Shows the tasks in the DAG: start, load_cleaned_data, auto_tag_topic, map_role_from_topic, generate_embeddi..., save_featured_out..., and end. All tasks are marked as successful (green bars).

Code Tab: Shows the DAG code.

Details Tab: Shows detailed information about the DAG.

Data Pipeline Demo

localhost:8080/dags/etl_preprocessing_dag

Dag etl_preprocessing_dag

Home Dags Assets Browse Admin Docs User

Options

etl_preprocessing_dag

Schedule Latest Run Next Run Owner Tags

Latest Dag Version v2

Trigger

Run Backfill Reparse Dag

start list_s3_folders extract_files basic_cleaning store_cleaned_data end

Last 24 hours 2025-05-11, 03:43:15 - 2025-05-12, 03:43:15

Last 3 Dag Runs

Run After	Duration (seconds)
2025-05-11, 05:00:00	5.97
2025-05-12, 03:39:47	0.58
2025-05-12, 03:41:38	-

```
graph TD; start((start)) --> listS3(list_s3_folders); listS3 --> extract(extract_files); extract --> clean1(basic_cleaning); clean1 --> clean2(store_cleaned_data); clean2 --> end((end))
```

Application prototype:

The image displays a web application prototype. On the left, there is a vibrant, isometric-style illustration of a large white AI head with a brain-like pattern inside, connected by yellow lines to several people working at desks or sitting in armchairs. The background is a light blue with binary code floating around. On the right, a separate window or card shows a login interface. The title 'AI Mock Interviewer' is at the top. Below it is a sub-instruction: 'Start here to prepare for your next interview with AI assistance'. There are two input fields: one for 'Enter your email' with a mail icon and another for 'Enter your password' with a lock icon. A 'Forgot password?' link is located below the password field. A large black 'Login' button is centered at the bottom. At the very bottom of the card, there is a link 'Don't have an account? Signup now'.

localhost:5173

AI Mock Interviewer

Start here to prepare for your next interview with AI assistance

Enter your email

Enter your password

[Forgot password?](#)

Login

[Don't have an account? Signup now](#)

localhost:5173/select-role

Customize Your Mock Interview

Select Your Company

Google Meta

Amazon OpenAI

Select Your Level

L2

Select Your Role

Data Analyst Data Scientist ML Engineer

Preview: Prep for Meta L2 Data Scientist

Start Interview

AI Mock Interview

Question: Can you explain the difference between supervised and unsupervised learning, and provide an example of each?

Type your answer here...

Submit Answer

AI Mock Interview

examples are relevant and well-chosen. You clearly explained the concepts and their applications, which demonstrates a solid understanding of the fundamentals. Well done!

Question: Can you explain what overfitting is and how it can be prevented in machine learning models?

You: overfitting happens when a model learns the training data too well including noise and random patterns which makes it perform poorly on new unseen data it can be prevented by using techniques like cross validation reducing model complexity pruning using regularization methods like L1 or L2 and increasing the amount of training data or using dropout in neural networks

Feedback: Your explanation of overfitting is clear and concise, effectively highlighting its impact on model performance. You also provided a solid range of prevention techniques, showcasing your understanding of practical approaches to improve model generalization. Great job!

Question: Can you describe what a confusion matrix is and how it can be useful in evaluating classification models?

You: a confusion matrix is a table used to evaluate the performance of a classification model by showing the counts of true positives true negatives false positives and false negatives it helps understand how well the model is performing beyond simple accuracy by revealing specific types of errors such as how many actual positives were missed or how many negatives were incorrectly labeled as positives

Feedback: Your definition of a confusion matrix is accurate and well-articulated, capturing its essential components and purpose. You effectively highlighted its importance in assessing model performance beyond just accuracy, which is crucial for nuanced evaluation in classification tasks. Excellent explanation!

Would you like to continue to the next question?

Yes, next question

No, end interview

Lama Fine-tuned prototype

```
[ ] question = generate_question("data science")
print(question)
```

👉 Q.1 Explain the Naïve Bayes classifier and discuss its advantages and disadvantages over other classifiers like Logistic Regression and Decision Trees.

```
▶ feedback, followup = get_feedback_and_followup(
    question=q,
    answer=a,
    domain="Data Science",
    **COMMON_KWARGS
)

print("Feedback:\n", feedback)
print("\nFollow-up Question:\n", followup)
```

👉 Feedback:
Your answer is clear and concise, and you've done a good job highlighting the key characteristics of the Naïve Bayes Classifier and comparing it with other classifiers. However, it would be helpful to pro-

Follow-up Question:
Can you explain how the Independence assumption is made in Naïves Bayes and what are its implications

Lama Sample prompt

```
] output_text = """You are an AI interview coach. Your task is:  
1. Evaluate the user's answer and provide helpful, constructive feedback.  
2. Ask a follow-up question - it can either continue the same topic or transition to a new domain from the list below.  
  
Available domains: Excel, Python, NoSQL, SQL, Python coding, Behavioral, Data Science, Probability, Machine Learning, Deep Learning, Statistics  
  
Input:  
Domain: SQL  
Question: What is the difference between WHERE and HAVING?  
User Answer: WHERE filters rows before grouping, HAVING filters after.  
  
Output format:  
Feedback: <feedback here>  
Follow-Up Question: <question here>  
  
Feedback: Great job! You're correct that WHERE is used for filtering rows before the aggregation functions are applied, while HAVING is used for filtering rows after the aggregation functions have been applied.
```

```
json.dump({  
    "messages": [  
        {"role": "system",  
            "content": f"You are a rigorous {domain} interviewer."},  
        {"role": "assistant",  
            "content": r["question"]},  
        {"role": "user",  
            "content": r["user_answer"]},  
        {"role": "assistant",  
            "content": f"{r['feedback']}\n\nFollow-up: {r['follow_up_question']}"}  
    ], fp)
```

```
def system_feedback(domain=None):  
    base = (  
        """You are an AI interviewer. Your task is:  
1. Evaluate the user's answer and provide helpful, constructive feedback.  
2. Ask a follow-up question - it can either continue the same topic or transition to a new domain from the list below.  
  
Available domains: Excel, Python, NoSQL, SQL, Python coding, Behavioral, Data Science, Probability, Machine Learning, Deep Learning, Statistics"""  
        "Do NOT repeat the candidate's answer verbatim."  
    )  
    if domain:  
        return base[0].format(domain=domain)  
    else:  
        return base[1]
```

```
def system_ask(domain=None):  
    if domain:  
        return f"You are a rigorous {domain} interviewer. Ask me a domain-specific question."  
    else:  
        return "You are a rigorous technical interviewer. Ask me a domain-specific question."
```

Mistral Fine-tuned prototype

```
q = generate_question("Data Science")
print("Q:", q)

a = input("A: ")

... Q: What is the difference between collinearity and multicollinearity? Provide an example to illustrate each concept.
A: 
```

The screenshot shows a terminal window with a dark background. At the top, there's a status bar with icons for file operations like up, down, copy, paste, and settings. The main area contains a code snippet in Python. It defines a function `generate_question` with the argument "Data Science". It then prints a question asking about the difference between collinearity and multicollinearity, followed by a prompt for an answer. Below the code, there's a user input field containing the text "A: ". The terminal window has a title bar at the top.

```
q = generate_question("Data Science")
print("Q:", q)

a = input("A: ")

... Q: What is the difference between collinearity and multicollinearity? Provide an example to illustrate each concept.
A: Collinearity occurs when two predictor variables are highly correlated (e.g., height and weight). Multicollinearity involves high correlation among three or more predictors (e.g., income, education, and job level in a salary model).
```

This screenshot shows the same terminal window after the user has provided an answer. The terminal now displays the AI's feedback, which explains the differences between collinearity and multicollinearity. The feedback states that collinearity occurs when two predictor variables are highly correlated (e.g., height and weight), while multicollinearity involves high correlation among three or more predictors (e.g., income, education, and job level in a salary model). The terminal window has a title bar at the top.

```
[34] feedback, followup = get_feedback_and_followup(q, a, domain="Data Science")
print("\n Feedback:\n", feedback)
print("\n Follow-up Question:\n", followup)
```

Feedback:
Great job! You've correctly identified that collinearity refers to a strong linear relationship between two predictors, while multicollinearity involves multiple predictors with strong linear relationships.

Follow-up Question:
In what ways does collinearity affect the coefficients of a regression model? How does multicollinearity affect the interpretation of the results? Can you provide an example of how collinearity or multicollinearity might impact a regression analysis?

Start coding or generate with AI.

The screenshot shows the terminal window again, this time displaying a detailed feedback message from the AI. The message starts with "Feedback:" and provides a clear explanation of the differences between collinearity and multicollinearity. It then asks a follow-up question about the effects of collinearity on regression coefficients and multicollinearity on regression interpretation. At the bottom of the window, there's a button labeled "Start coding or generate with AI." The terminal window has a title bar at the top.

Company, Round & Role Specific Data

	A	B	C	D	E	F	G	H	I	J
1	Company	Role	Tags	Round Number	Round Name	Topic Questions				
2	Meta	Machine Learning Engineer	Behavioral,Machinelearning,Programminglanguages,Frameworks	Round 1	HR Interview	Why do you want to join Meta?				
3	Meta	Machine Learning Engineer	Behavioral,Machinelearning,Programminglanguages,Frameworks	Round 1	HR Interview	Why do you think you will be a good fit for the role?				
4	Meta	Machine Learning Engineer	Behavioral,Machinelearning,Programminglanguages,Frameworks	Round 1	HR Interview	What responsibilities do you expect to have from your job at Meta?				
5	Meta	Machine Learning Engineer	Behavioral,Machinelearning,Programminglanguages,Frameworks	Round 1	HR Interview	What are your areas of expertise in machine learning, and how have you developed those skills?				
6	Meta	Machine Learning Engineer	Behavioral,Machinelearning,Programminglanguages,Frameworks	Round 1	HR Interview	What programming languages are you comfortable with, and how have you used them in your previous projects?				
7	Meta	Machine Learning Engineer	Behavioral,Machinelearning,Programminglanguages,Frameworks	Round 1	HR Interview	What machine learning frameworks have you used, and how did you find them helpful in your projects?				
8	Meta	Machine Learning Engineer	Behavioral,Machinelearning,Programminglanguages,Frameworks	Round 1	HR Interview	Have you worked with large datasets?				
9	Meta	Machine Learning Engineer	Behavioral,Machinelearning,Programminglanguages,Frameworks	Round 1	HR Interview	If so, how did you handle data preprocessing and ensure data quality?				
10	Meta	Machine Learning Engineer	Behavioral,Machinelearning,Programminglanguages,Frameworks	Round 1	HR Interview	Have you implemented any machine learning algorithms from scratch?				
11	Meta	Machine Learning Engineer	Behavioral,Machinelearning,Programminglanguages,Frameworks	Round 1	HR Interview	If so, can you provide an example and explain your thought process?				
12	Meta	Machine Learning Engineer	Coding,Datastructures,Algorithms,Bigo,Python	Round 2	Technical Screen	Can you walk me through how you would approach solving a coding problem?				
13	Meta	Machine Learning Engineer	Coding,Datastructures,Algorithms,Bigo,Python	Round 2	Technical Screen	How would you ensure that your code is optimized and free of logic flaws?				
14	Meta	Machine Learning Engineer	Coding,Datastructures,Algorithms,Bigo,Python	Round 2	Technical Screen	Can you provide an example of a challenging problem you solved with a creative solution?				
15	Meta	Machine Learning Engineer	Coding,Datastructures,Algorithms,Bigo,Python	Round 2	Technical Screen	What are your strengths when it comes to data structures and algorithms?				
16	Meta	Machine Learning Engineer	Coding,Datastructures,Algorithms,Bigo,Python	Round 2	Technical Screen	How do you handle ambiguity in problem-solving?				
17	Meta	Machine Learning Engineer	Coding,Datastructures,Algorithms,Bigo,Python	Round 2	Technical Screen	Can you explain the concept of Big O notation and its significance in coding?				
18	Meta	Machine Learning Engineer	Coding,Leetcode,Datastructures,Debugging	Round 3	Onsite Interview - Coding	Design a data structure that supports insert, delete, search and get random in constant time., Find the nth node in a linked list.				
19	Meta	Machine Learning Engineer	Coding,Leetcode,Datastructures,Debugging	Round 3	Onsite Interview - Coding	Implement a function to construct a binary tree., Write a code for the dot product of two sparse vectors.				
20	Meta	Machine Learning Engineer	Coding,Leetcode,Datastructures,Debugging	Round 3	Onsite Interview - Coding	Given an infinite chessboard, find the shortest distance for a knight to move from position A to position B.				
21	Meta	Machine Learning Engineer	Systemdesign,Scalability,Architecture,Communication	Round 3	Onsite Interview - System Design	Design a scheduler for distributed cron jobs., Build a recommendation based engine., Design ML system architecture.				
22	Meta	Machine Learning Engineer	Mlsystemdesign,Modeling,Deployment,Featureengineering	Round 3	Onsite Interview - ML Design	What are the roles of batch normalization and dropout in training deep neural networks?				

Creating Embeddings and Storing in ChromDB

```
# Step 2: Load Excel
import pandas as pd
filename = list(uploaded.keys())[0]
df = pd.read_excel(filename)

# Step 3: Clean whitespace in key columns
for col in ['Company', 'Role', 'Round Name', 'Topic Questions']:
    df[col] = df[col].astype(str).str.strip()

# Step 4: Create combined text for embedding
df['combined_text'] = df.apply(
    lambda row: f"{row['Company']} - {row['Role']} - {row['Round Name']} - {row['Topic Questions']}",
    axis=1
)

# Step 5: Install and load embedding model
!pip install -q sentence-transformers
from sentence_transformers import SentenceTransformer
import json

model = SentenceTransformer('all-MiniLM-L6-v2')

# Step 6: Generate embeddings
df['embedding'] = df['combined_text'].apply(lambda x: model.encode(x).tolist())

# Step 7: Drop combined text
df.drop(columns=['combined_text'], inplace=True)
```

```
[ ] # Step 3: Initialize Chroma PersistentClient
persist_dir = "./chroma_db"
os.makedirs(persist_dir, exist_ok=True)

from chromadb import PersistentClient
client = PersistentClient(path=persist_dir)

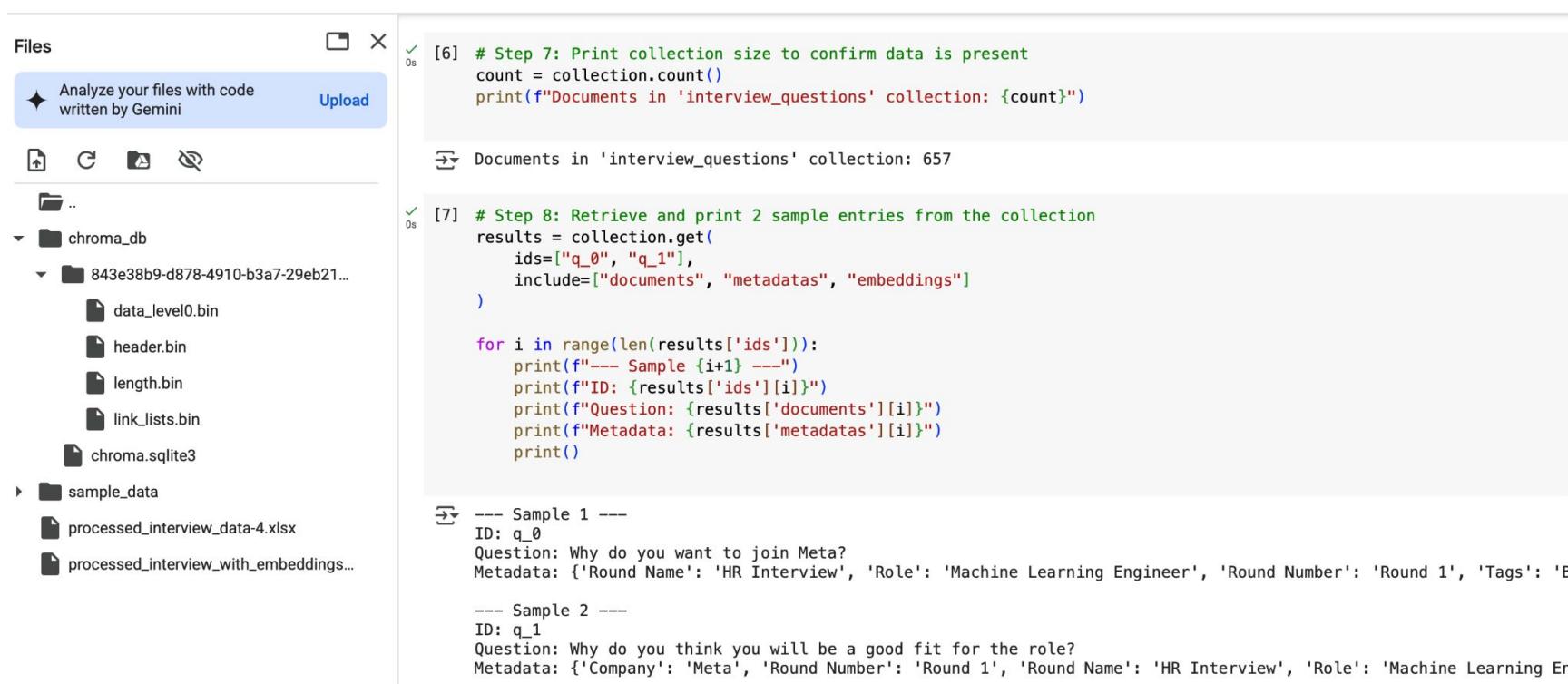
# Step 4: Create or get a collection
collection = client.get_or_create_collection(name="interview_questions")

# Step 5: Load JSONL file
records = []
with open('processed_interview_with_embeddings.jsonl', 'r') as f:
    for line in f:
        record = json.loads(line)
        records.append(record)

# Step 6: Prepare and add data
texts = [r['Topic Questions'] for r in records]
embeddings = [r['embedding'] for r in records]
metadata = [{k: v for k, v in r.items() if k not in ['embedding', 'Topic Questions']} for r in records]

collection.add(
    documents=texts,
    embeddings=embeddings,
    metadatas=metadata,
    ids=[f"q_{i}" for i in range(len(texts))]
)
```

Data Stored in ChromaDB



The screenshot shows a terminal window with a file browser interface on the left. The terminal displays two steps of Python code for interacting with a 'interview_questions' collection in ChromaDB.

Terminal Output:

```
[6] # Step 7: Print collection size to confirm data is present
count = collection.count()
print(f"Documents in 'interview_questions' collection: {count}")

→ Documents in 'interview_questions' collection: 657

[7] # Step 8: Retrieve and print 2 sample entries from the collection
results = collection.get(
    ids=["q_0", "q_1"],
    include=["documents", "metadata", "embeddings"]
)

for i in range(len(results['ids'])):
    print(f"--- Sample {i+1} ---")
    print(f"ID: {results['ids'][i]}")
    print(f"Question: {results['documents'][i]}")
    print(f"Metadata: {results['metadata'][i]}")
    print()

→ --- Sample 1 ---
ID: q_0
Question: Why do you want to join Meta?
Metadata: {'Round Name': 'HR Interview', 'Role': 'Machine Learning Engineer', 'Round Number': 'Round 1', 'Tags': 'B

--- Sample 2 ---
ID: q_1
Question: Why do you think you will be a good fit for the role?
Metadata: {'Company': 'Meta', 'Round Number': 'Round 1', 'Round Name': 'HR Interview', 'Role': 'Machine Learning En
```

File Browser:

- Files
- Analyze your files with code written by Gemini
- Upload
- ...
- chroma_db
 - 843e38b9-d878-4910-b3a7-29eb21...
 - data_level0.bin
 - header.bin
 - length.bin
 - link_lists.bin
 - chroma.sqlite3
- sample_data
 - processed_interview_data-4.xlsx
 - processed_interview_with_embeddings...

RAG + NER

```
[ ] # Read and preview JSONL file in Colab
import json

with open('processed_interview_with_embeddings.jsonl', 'r') as f:
    lines = f.readlines()

# Parse first few JSON objects
for i, line in enumerate(lines[:5]):
    data = json.loads(line)
    print(f"\n--- Record {i+1} ---")
    print(json.dumps(data, indent=2))
```



```
--- Record 1 ---
{
  "Company": "Meta",
  "Role": "Machine Learning Engineer",
  "Tags": "Behavioral,Machinelearning,Programminglanguages,Frameworks",
  "Round Number": "Round 1",
  "Round Name": "HR Interview",
  "Topic Questions": "Why do you want to join Meta?",
  "embedding": [
    -0.028817638754844666,
    0.0369221568107605,
    0.01961103081703186,
    0.04577894136309624,
    0.15243588387966156,
    -0.045289553701877594,
    0.041193582117557526,
```

RAG Embeddings

```
[ ] # 2. Metadata extractor using Gemini prompt
def extract_metadata_from_query():
    user_query = input("Enter your interview query: ")
    prompt = f"""
Extract the following information from the sentence below:

Sentence: "{user_query}"

Return a JSON object with:
- Company
- Role
- Round Number

Format:
{{"
  "Company": "...",
  "Role": "...",
  "Round Number": ...
}}
"""

try:
    response = model.generate_content(prompt)
    response_json = response.text.strip()
    match = re.search(r'\{.*\}', response_json, re.DOTALL)
    if match:
        metadata = json.loads(match.group())
        print(" Extracted Metadata:", metadata)
        return metadata
    else:
        print("Could not find valid JSON in response.")
        return None

```

Gemini Prompt for NER

RAG Results

```
▶ run_full_pipeline()

➡ Enter your interview query: Could you help me practice for Data Scientist Round 3 interview for Meta
Extracted Metadata: {'Company': 'Meta', 'Role': 'Data Scientist', 'Round Number': '3'}
```

🔍 Querying: interview questions for Data Scientist Round 3

--- Match 1 ---

Question: Which data sets would you use to answer specific product-related questions?
Metadata: {'Round Name': 'Analytical Reasoning Interview', 'Role': 'Data Scientist', 'Tags': 'Behavioural,Statistics,ProductCase,ExperimentDesign,BiasDetection,DataIntegration'}

--- Match 2 ---

Question: Given a dataset on user engagement, how would you structure a solution to improve user interaction on our platform?
Metadata: {'Tags': 'Behavioural,Statistics,ProductCase,Sql,Coding,EdgeCases,DataAdaptation,CodeOptimization,Communication,Metrics', 'Round Number': '3'}

--- Match 3 ---

Question: How do you draw meaningful conclusions from a dataset?
Metadata: {'Round Name': 'Analytical Reasoning Interview', 'Tags': 'Behavioural,Statistics,ProductCase,ExperimentDesign,BiasDetection,DataIntegration'}

--- Match 4 ---

Question: How do you integrate information from various sources into a cohesive insight?
Metadata: {'Company': 'Meta', 'Round Name': 'Analytical Reasoning Interview', 'Role': 'Data Scientist', 'Tags': 'Behavioural,Statistics,ProductCase,ExperimentDesign,BiasDetection,DataIntegration'}

--- Match 5 ---

Question: Code a solution to analyze customer behavior data and identify patterns that could inform marketing strategies., Consider a scenario where we want to predict future purchases based on past buying habits.
Metadata: {'Tags': 'Behavioural,Statistics,ProductCase,Sql,Coding,EdgeCases,DataAdaptation,CodeOptimization,Communication,Metrics', 'Role': 'Data Scientist'}

Results from RAG



Q and A?