Product Management Intern - Task

Tasks will be checked on a first come first serve basis
Number of questions: 2

Product Challenge

About Humanness by JoshTalks

JoshTalksAI (Humanness) is building gold-standard AI datasets across modalities — speech, image, and text — that help train and fine-tune advanced AI models such as Automatic Speech Recognition (ASR) and Vision-Language (VL) systems.

We operate like a **data research lab**, creating high-quality, accurately labeled datasets that improve the performance of global AI models.

Why ASR and High-Quality Datasets Matter

What is ASR (Automatic Speech Recognition)?

ASR models convert **spoken language (audio)** into **written text** — they are the backbone of technologies like:

- Voice assistants (e.g., Alexa, Siri, Google Assistant)
- Meeting transcription tools
- Customer-service bots and call analytics
- Accessibility tools for captioning and dictation
- Voice-based search and commands

These models depend on **training data** — pairs of audio and text that teach the AI what words sound like across languages, accents, and tones.

Why Datasets Matter

To "teach" an ASR model a language, it needs thousands of hours of paired examples:

Audio File	Transcription	Language	Duration
`speaker_001_segment_01.wav`	"मैं कल स्कूल जाऊँगा।"	Hindi	6.5 sec
`speaker_024_segment_15.wav`	"This is my favorite restaurant."	English	7.2 sec
`speaker_310_segment_08.wav`	"నేను ఈ రోజు మార్కెట్‌కి వెళ్తాను."	Telugu	8.1 sec

Each pair is like a teacher-student example.
The model learns *what words sound like* in different accents, tones, and background conditions.

When these datasets are inaccurate — e.g., transcription errors, background noise, or wrong spellings — the AI model's **Word Error Rate (WER)** increases.

Our goal at JoshTalksAI is to create **datasets so clean** that they achieve **WERs as low as possible (98–99% word-level accuracy)**.

The Role of Gold-Standard Datasets

An ASR model's quality is measured by its **Word Error Rate (WER)** — the percentage of words it gets wrong.
Lower WER means higher accuracy.

To achieve that, models need **large, clean, and accurate datasets** where:

- Audio is recorded in good quality.
- Transcriptions match exactly what's spoken.

- Noise and irrelevant data are removed.
- Annotations are reviewed multiple times for correctness.

Every misheard or mislabeled word increases WER — so **data quality directly determines model performance**.

Beyond Speech: Vision and Multimodal Datasets

While ASR deals with speech, AI models also need to understand **images, scenes, and cultural symbols**.
For example:

- An AI may recognize the Eiffel Tower instantly but fail to recognize a **Durga Puja pandal** or a **Rath Yatra procession**.
- To make AI systems culturally inclusive, they need **image datasets** that represent India's people, landscapes, festivals, and daily life — across **every district**.

This is why, beyond speech, **JoshTalksAI** also works on **image and video datasets** that reflect India's real-world diversity.

TASK 1: Design a Platform to Collect Images + Descriptions from All Villages of India

Context

JoshTalksAI also builds **vision and multimodal datasets**. Today's AI models can identify the Eiffel Tower but may not recognize **a Durga Puja pandal** or a **handpump in rural Rohtak**.

To teach AI about India's cultural and geographic richness, we aim to collect:

1000 images per village across all Indian districts, each with a short text description.

Each sample helps train or evaluate vision-language models (like CLIP or GPT-4V) on Indian contexts.

Objective

Design a simple, scalable product that enables contributors across India to **capture, describe, and submit images** from their districts — and allows internal teams to **verify and monitor** coverage.The focus is on **clarity, inclusivity, and data integrity** rather than visual polish.

Directions

1 · Understand the Use Case

- Review why culturally rich, location-verified imagery is important for AI.
- Think about who contributes these images (field agents, NGO partners, individual volunteers) and what limitations they face: low connectivity, varied devices, multilingual input, minimal training.

2 · Define User Roles and Goals

- Identify the core user types (contributors vs reviewers/admins).
- Clarify what each needs to accomplish:
    - Contributors → capture/upload image + add description + submit.
    - Reviewers/Admins → verify quality and district coverage.

3 · Design the Contributor Flow

- Map the end-to-end experience:

    - Accessing the product (app / mobile web).
    - Selecting or confirming state + district.
    - Capturing or uploading a photo.
    - Writing a short caption or description.
    - Reviewing and submitting.
- Consider what validations or confirmations are required before submission.
- Keep the process minimal, mobile-friendly, and resilient in low-network areas.

4 · Account for Edge Scenarios

- Weak or no internet → how are drafts saved or synced later?
- Very short or missing descriptions → how do you nudge for meaningful input?
- Wrong district → how does the user correct or confirm location?
- Low-quality or irrelevant images → how are they flagged or re-checked?

5 · Design the Admin / Verification View

- Outline a simple internal interface that shows:
    - District-wise coverage and progress.
    - Filters by state, district, date, or contributor type.
    - Image + caption pairs for manual verification.
    - Indicators for approved, rejected, or pending submissions.
- Focus on clarity, searchability, and quick validation.

6 · Summarize Your Approach

Prepare a concise write-up covering:

- The problem you're solving and the assumptions you made about users.
- The flow and design principles guiding your solution (simplicity, inclusivity, accountability).
- The outcomes you expect if the platform is adopted.
- Key metrics you'd track to measure success

Deliverables

1. **PRD (Product Requirements Document)** outlining:
    - Problem & goals.
    - User types and core flows.
    - MVP features.
    - Success metrics.
2. **Figma Designs (or similar):**
    - Contributor flow (select district → capture/upload → describe → submit).
    - Admin dashboard (coverage overview + image gallery with captions).

The goal is to create a product that enables India-wide image collection efficiently and responsibly — helping AI models see and understand the country through authentic, well-described visuals.

Task 2

Background
As shared above - The newest and the fastest growing division of Josh Talks is working to create datasets to train AI models. This data division is building India's largest-scale, high-quality multilingual voice and vision datasets to train and improve AI models. As you may have seen, AI tools like ChatGPT and others are becoming a big part of our daily lives. Behind the scenes, the quality of the data used to train these models plays a huge role in how well they work - which is why high-quality training data has become more important than ever!

For example, our data engine collects millions of hours of Indian speech, covering 25+ languages, diverse dialects, socio-economic backgrounds, emotional tones, and ambient environments. All data is subjected to **human-in-the-loop quality checks**, ensuring annotation error remains below 2% and delivering enterprise-grade accuracy.

These human review tasks are executed on our in-house platform, Josh Jobs, where we offer freelance opportunities to individuals across India.

The two most common roles include:

- **Recording Tasks:** Contributors submit voice recordings in their native languages, helping us capture 10,000+ hours of audio daily across multiple languages.
- **Transcription & Quality Check Tasks:** Contributors listen to short audio clips in regional languages and transcribe them into accurate text. We currently have over 20,000 daily active users engaged in transcription.

This task is a sample problem statement from a transcription and quality check task.

Here's a simplified look at how transcription works on our platform:

Here's how transcription works:

- Audio Processing: We have audio chunks (collected from recording) that need accurate text transcriptions
- AI First Pass: We use Whisper AI to create an initial transcription of each audio chunk
- Human Review: Transcribers log into our platform, listen to the audio, and see Whisper's text
- Quality Improvement: They edit Whisper's transcription to fix any errors and create the final, accurate version
- Payment: Transcribers get paid based on the number of accurate hours of audio they complete

The problem is that some transcribers are doing poor quality work - either rushing through tasks, making minimal effort, or not actually listening to the audio properly. This hurts our final product quality and wastes money on bad work.

Your Task

Part-I
Identify these low-quality transcribers using data patterns. You have access to data (here) of some users and their completed transcription tasks. Each row contains:

- `user_id`: Which person did the work
- `recording_url`: The audio file they worked on
- `whisper_text`: What the AI originally wrote
- `user_text`: What the transcriber submitted as their final version
- `is_edited:` Yes if `user_text` is different from `whisper_text` (i.e., user made edits).
  `duration`: How long the audio clip was (in seconds)
- `time_taken_by_user`: How long the transcriber spent on this task (in seconds)
- `segment_character_per_second`: : Non-space characters in `user_text` divided by `time_taken_by_user` (in seconds).

a) Analyze this data to identify at least 2 warning signs that someone might be doing poor quality work.
b) For each pattern you identify, explain:
    - What it tells us: Why does this behavior suggest poor quality?
    - How to measure it: What would you calculate from the data to spot this pattern?
    - Red flag thresholds: At what point should we be concerned?
    - False alarms: When might good transcribers show this pattern innocently?

Part-II
Based on your analysis in part-I, design a practical system to automatically spot problematic transcribers on the platform itself before they do too much damage

Create clear, actionable recommendations that you could give to our engineering team. Focus on the transcription accuracy and transcriber experience, not the technical coding. Keep in mind that the logic/metric you share will be used to block transcriber's accounts so make sure you are 100% sure about the logic. Your recommendations should look something like this - If a transcriber exhibits X,Y or/and Z behaviour; block the user.