Problem statement: *Eywa SDE Intern*

Build a Telegram YouTube Summarizer & Q&A Bot

Using OpenClaw

---

1. Objective

Set up **OpenClaw** locally and build a Telegram bot that:

1. Accepts a YouTube link
2. Fetches the video transcript
3. Generates a clear, structured summary
4. Allows users to ask questions about the video
5. Supports English **and at least one Indian language** (e.g., Hindi, Tamil, Kannada, etc.)

This is a **business-focused assignment**. Implementation approach is your choice.

---

2. Business Requirement

We want to build a smart assistant that helps users:

- Understand long YouTube videos quickly
- Extract key insights
- Ask contextual questions
- Consume content in their preferred language

The system should behave like a personal AI research assistant for YouTube.

---

3. Core User Flow

Step 1 — User Sends YouTube Link

User:

https://youtube.com/watch?v=XXXXX

Bot responds with:

🎥 Video Title

📌 5 Key Points
⏱ Important Timestamps
🧠 Core Takeaway

Step 2 — User Asks Questions

User:

What did he say about pricing?

Bot:

- Answers clearly using video context
- If not found in transcript, responds:

  This topic is not covered in the video.

Step 3 — Multi-language Support

User can request:

Summarize in Hindi

or

Explain in Kannada

Bot must support:

- English (default)
- At least **one additional Indian language**

Language support may include:

- Summary

- Q&A responses
- Simplified explanations

---

4. Functional Requirements

4.1 Setup

- Install OpenClaw locally
- Create Telegram bot via BotFather
- Connect Telegram with OpenClaw skill

Deliverable: Bot responds to basic messages.

4.2 Transcript Retrieval

Use a public transcript method such as:

- **youtube-transcript-api**

System must:

- Handle invalid links
- Handle missing transcripts
- Handle long transcripts
- Gracefully manage errors

4.3 Summary Requirements

The summary must:

- Be structured (not paragraph dump)
- Include:
    - Key points
    - Timestamps
    - Core insight

- Be concise but meaningful

4.4 Q&A Requirements

- User can ask multiple follow-up questions
- Answers must be grounded in the transcript
- No hallucinations
- Must handle multiple users simultaneously

How you implement context handling, storage, retrieval, chunking, etc. is your architectural decision.

4.5 Language Requirement

The bot must:

- Detect or accept requested language
- Generate responses in:
    - English
    - At least one Indian language

Examples:

- Hindi
- Tamil
- Telugu
- Kannada
- Marathi

Implementation approach is up to you (translation layer, multilingual model, etc.).

---

5. Architecture Freedom

You are free to design:

- How transcript is stored
- How context is managed
- How questions are answered

- Whether to use chunking, embeddings, caching, etc.

However, you must document your architectural decisions in README.

---

6. Evaluation Criteria

Category	Weight
End-to-end functionality	30%
Summary quality	20%
Q&A accuracy	20%
Multi-language support	15%
Code quality & structure	10%
Error handling	5%

---

7. Bonus (Optional)

- Smart caching of transcripts
- Cost optimization (token efficiency)
- Clean session management
- Add commands like:
  - `/summary`
  - `/deepdive`
  - `/actionpoints`

---

8. Edge Cases to Handle

- Invalid YouTube URL
- No transcript available
- Non-English transcript
- Very long video

- Rate limiting

---

9. Submission Requirements

- GitHub repository
- README including:
  - Setup steps
  - Architecture explanation
  - Design trade-offs
- Demo video (3–5 minutes)
- Example screenshots - include in the readme

---

10. Timeline

Expected completion time: **3–4 days**
