Week 1 Report –

Video Analysis Application Development

1. Project Scopes and Goals

Scope

This project aims to design and develop an AI/ML-powered application that can automatically analyze long-form YouTube product review videos and extract key highlight segments. The focus is on three critical parts of a typical review:

- **Product Unboxing**
- **Feature Demonstrations**
- **Final Verdict/Conclusion**

The system will combine speech recognition, scene detection, and visual/audio cues to identify these segments and output short clips (30–60 seconds each).

Goals

- Improve video accessibility and engagement by automatically generating short highlight clips.
- Reduce manual editing time for content creators.
- Provide reusable, configurable tools that can be adapted to different video formats and topics.
- Demonstrate integration of NLP (speech transcription), CV (visual detection), and AV (audio analysis) into a unified pipeline.

2. Existing Applications and Techniques

Several applications and research work already tackle aspects of highlight extraction, video summarization, and scene segmentation:

- **YouTube Chapters & Auto-generated Highlights**: YouTube provides automatic chaptering based on metadata and captions, but it is limited in customization and control.
- **Sports Highlight Generation**: Uses crowd noise, commentator excitement, and visual changes to extract highlights. Techniques include audio energy detection, shot boundary detection, and keyword spotting.
- **Lecture/Conference Summarization Tools**: Rely on ASR-based keyword spotting and slide/scene change analysis.

- **Video Editing Assistants (e.g., Pictory, Wisecut, Magisto)**: Commercial AI-based video summarization tools that use transcript analysis, emotion detection, and template-based editing.

Techniques applicable to our project:

- **Automatic Speech Recognition (ASR)** → Extract transcripts with timestamps for keyword-based detection.
- **Keyword & Semantic Analysis** → Identify "unboxing", "demo", "final thoughts" cues.
- **Scene/Shot Detection** → Detect visual transitions like box opening or camera close-ups.
- **Object/Action Recognition** → Identify objects (box, device) or actions (hands, demo).
- **Audio Cues** → Silence detection, sound of packaging, or tonal emphasis in final verdict.

3. Proposed Architecture

The system will follow a modular pipeline

1. **Video Input Layer**
   - Input from YouTube URL or local MP4 file.
   - Download using `yt-dlp`.
2. **Preprocessing**
   - Extract audio track using `ffmpeg`.
   - Normalize video for frame sampling and scene detection.
3. **Analysis Modules**
   - ➢ **Speech Recognition**: Use Whisper/WhisperX for transcripts with timestamps.
   - ➢ **Keyword Detection**: Search transcripts for event-specific terms.
   - ➢ **Scene Detection**: Apply PySceneDetect for shot boundary detection.
   - ➢ **Visual Analysis**: Use YOLO or lightweight classifiers to detect product, box, and demo scenes.
   - ➢ **Audio Analysis**: Silence and energy-based cues for emphasis.
4. **Event Fusion & Decision Layer**
   - Combine transcript cues, scene changes, and visual/audio evidence.
   - Generate event timestamps with confidence scores.
5. **Clip Extraction**
   - Use ffmpeg to cut 30–60 second clips around detected events.

- Save metadata (event type, timestamps, confidence).

6. **Output**

- Highlight clips ready for upload to YouTube Shorts/Reels.
- JSON file for metadata and indexing.