Personalized Educational Assistant for Students with Learning Disabilities

Operating systems-23AID213

Mentor:
-Ms. Pooja Gowda
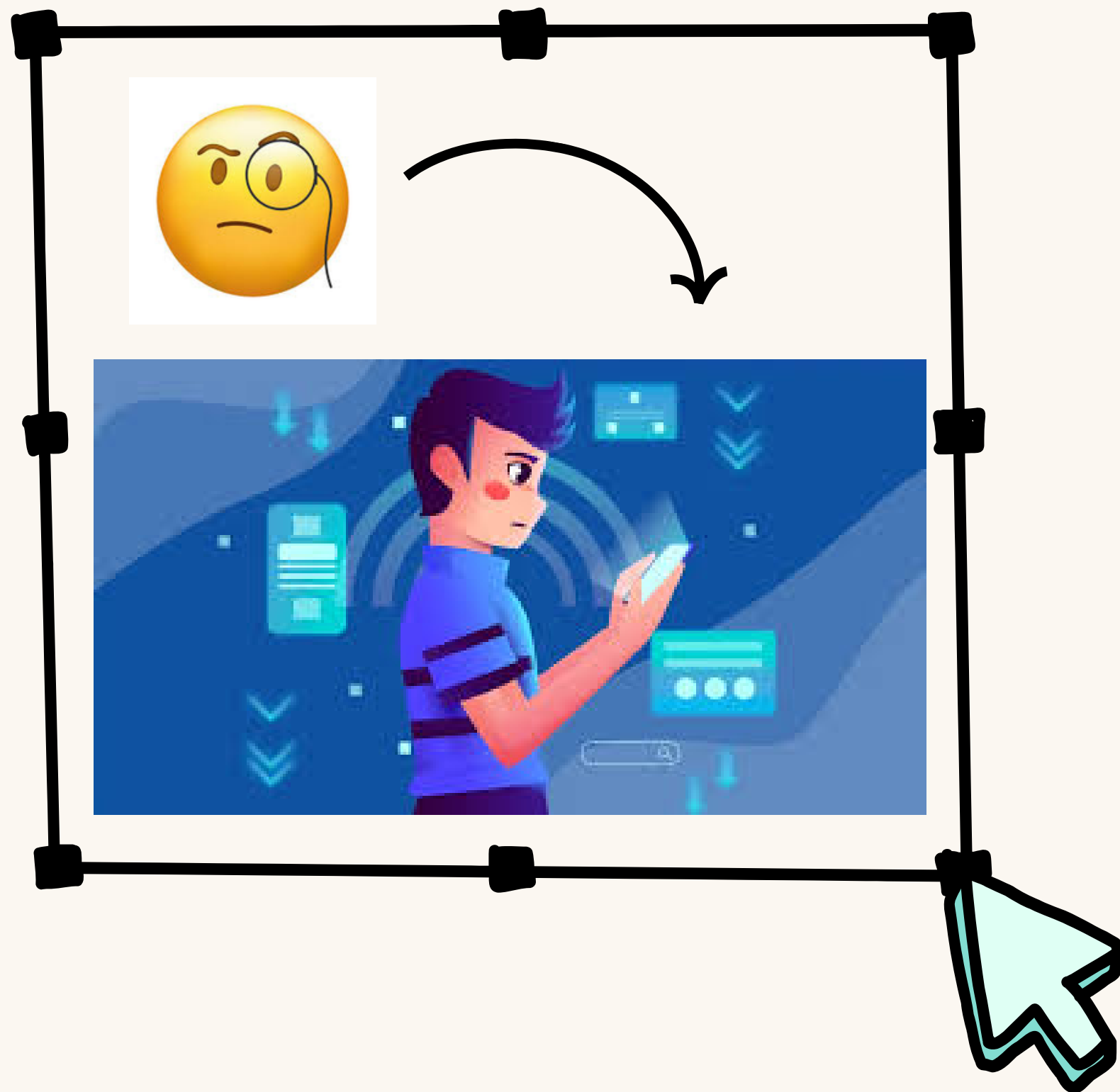
Presented By:
-Amara Pranav (03)
-Joshika Somisetty (19)
-Koduri Lakshmi Vinugna (26)

Agenda

1. Introduction
2. Motivation
3. Literature Survey
4. Problem Statement
5. Project Objectives
6. Methodology
7. Results
8. Conclusion
9. Future work

Introduction

- Many students face learning challenges such as:
  1. Dyslexia
  2. ADHD(Attention-Deficit/Hyperactivity Disorder)
  3. Cognitive impairments
- These issues make traditional educational methods ineffective.
- Need for adaptive, real-time support to assist diverse learners.
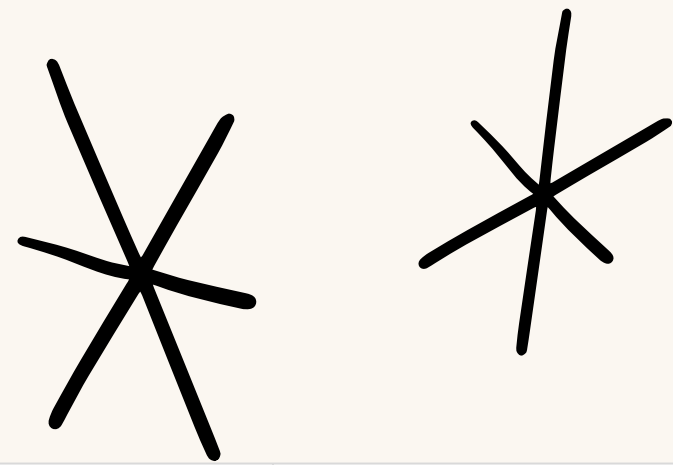- Integrates OS-level features for smarter interaction.
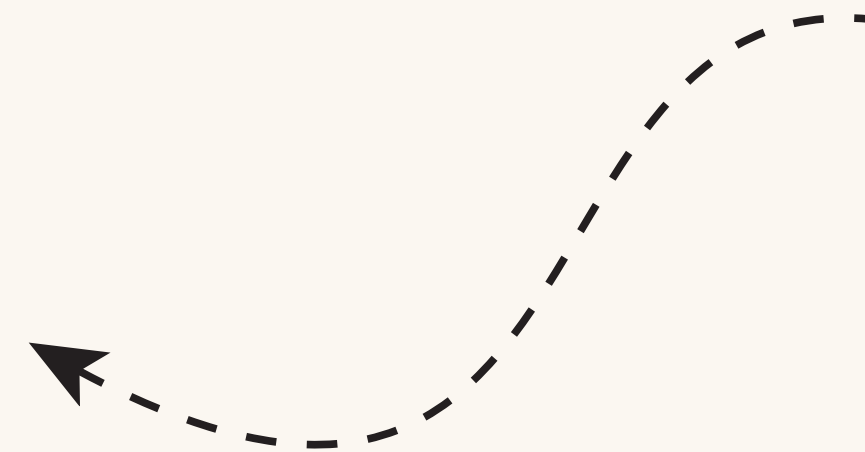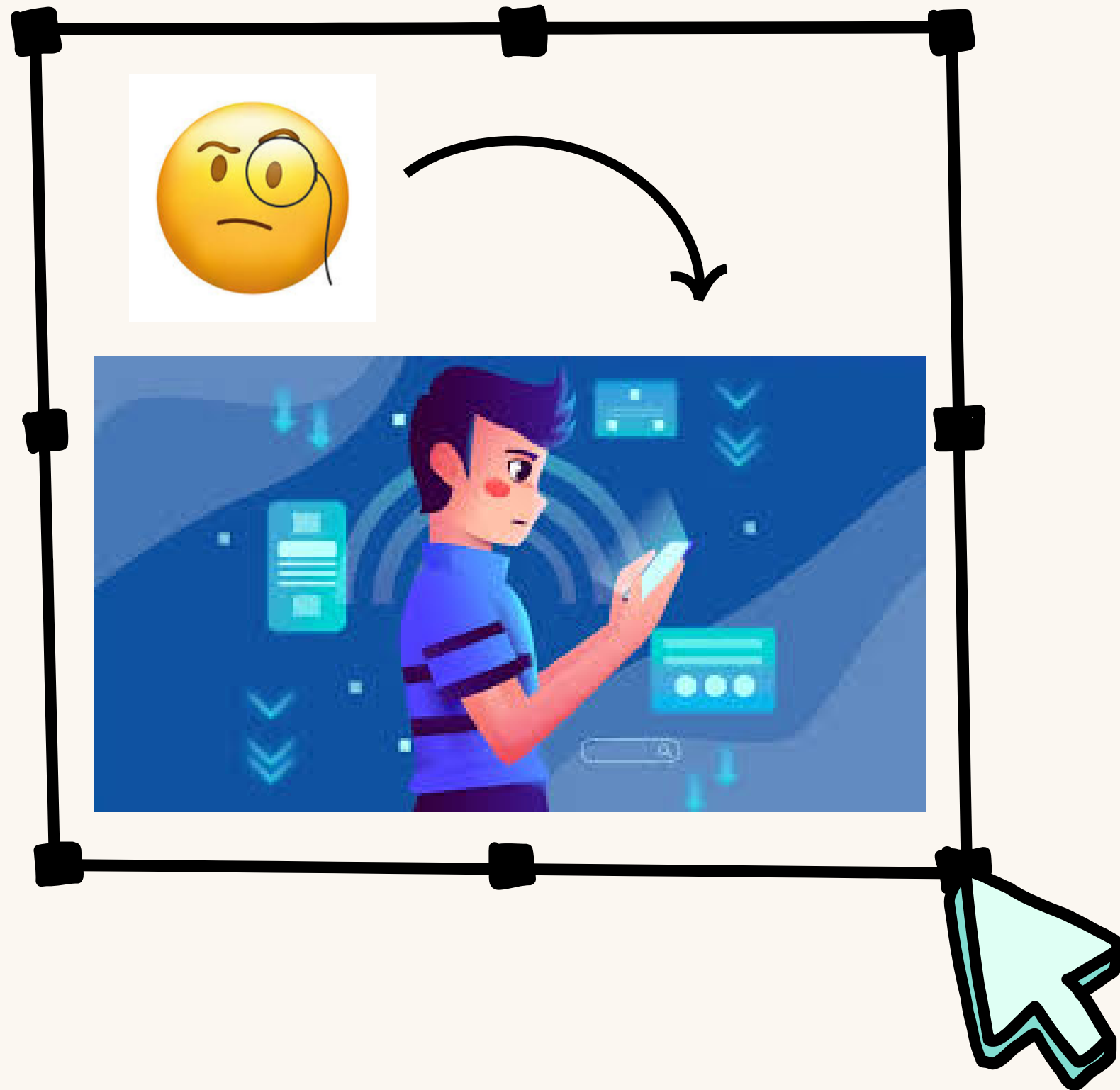
Motivation

- Over 10% of students globally are affected by learning disabilities.
- Existing tools are not tailored to individual learning needs.
- Lack of real-time adaptability.
- OS concepts enable:
  1. Real-time processing
  2. Efficient resource handling
  3. Personalized experiences
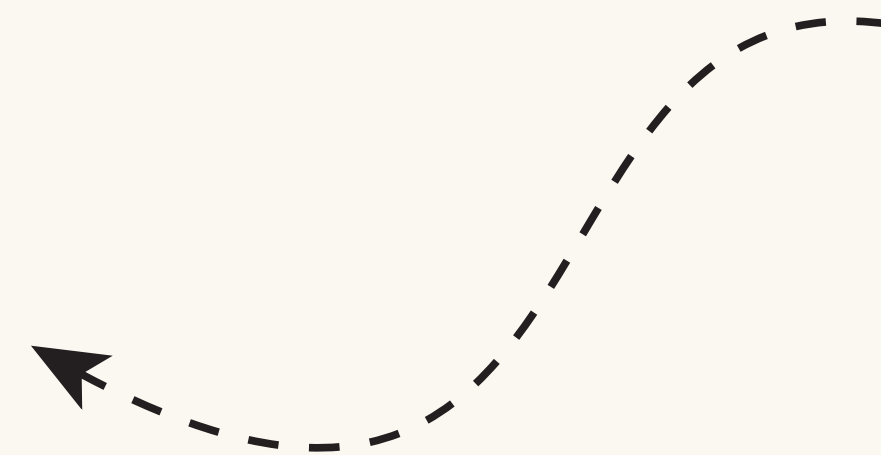
Literature survey

| Paper	Title	Year
1	Leveraging AI Technologies for Personalized Learning Support in Dyslexic Students	2022
2	AI-Powered Personalized Learning Assistant	2024
3	Adaptive E-learning Platform for Primary School Children with Visual Impairment	2023
4	AI-Enabled Intelligent Assistant for Personalized and Adaptive Learning in Higher Education	2024
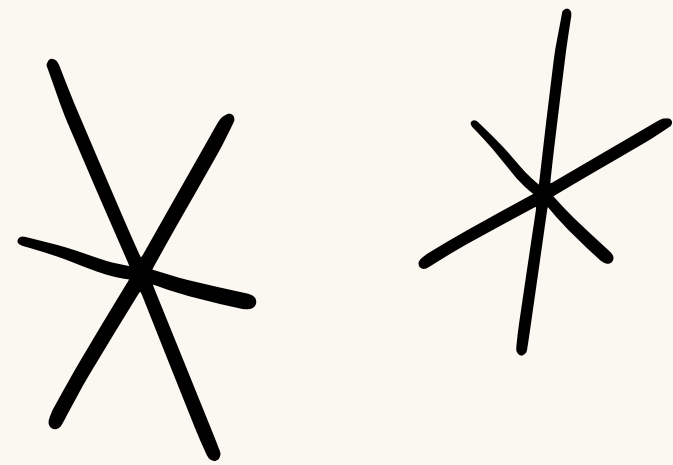
Literature survey

Key Technologies	OS Concepts Mentioned
NLP, ML, Speech Recognition, Text-to-Speech	Multitasking, Resource Management, Real-Time Processing
React, Node.js, Flask, TensorFlow.js, ML, NLP	Cloud-based delivery, Implicit scalability and system optimization
Voice-based interaction, Learning tech integration	Implicit: OS Accessibility Features, Voice-Interaction Support
AI, NLP, Interactive Platforms	Concurrency, Memory Management, API Integration, Security & Privacy

▶ Problem Statement

Students with learning disabilities often struggle to process and retain information effectively. A Virtual Educational Assistant can convert lectures into structured notes in real-time, improving comprehension and retention.

How OS Concepts Solve the Problem
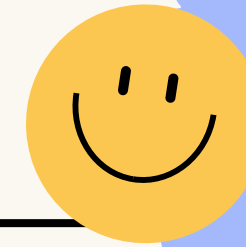
OS Concept	Application in Project
Multithreading	Parallel execution of face tracking and speech summarization
Synchronization	Thread-safe printing and shared state updates
Event Handling	Global signal to terminate both threads gracefully
Resource Management	Efficient use of camera/audio input without performance lag
I/O Management	Real-time processing of camera, mic, and user keyboard inputs
Memory Management	Memory is managed via garbage collection, fixed buffers, and resource cleanup.

Project Objectives

- Developing a real-time virtual educational assistant for students with learning disabilities.
- Monitor attention levels using webcam-based distraction detection.
- Transcribe and summarize spoken lectures using AI-based speech-to-text model.
- Providing alerts to help students stay focused during learning sessions.
- Ensuring to implement all possible OS concepts.

Methodology

1. System Initialization

- Import essential libraries for:
1. Computer Vision (cv2, mediapipe, numpy)
2. Speech Processing (speech_recognition, pyttsx3, transformers)
3. Thread Management (threading)
- Create shared resources:
4. Lock for safe console output
5. Event to signal program termination

3. Speech Transcription & Summarization (Thread 2)

- Speech Recognition
- Transcription: Uses Google Speech Recognition to convert voice to text.
- Summarization Trigger
- Summarization & Keyword Extraction:
1. Uses facebook/bart-large-cnn model to summarize transcribed text
2. Identifies key points and most frequent words for better retention

2. Distraction Detection (Thread 1)

- Camera Input: Captures live video feed using OpenCV.
- Face Mesh Detection: Uses MediaPipe to detect facial landmarks.
- Distraction Logic:
1. Face Turn Detection
2. Eye Closure Detection
- Distraction Alert: If a distraction persists:
- Trigger audio feedback using pyttsx3
- Display alert and counter on screen

4. Parallel Execution and Termination

- Both modules run concurrently using threads.
- A shared termination flag (terminate_event) ensures:
1. Clean shutdown of all processes once summarization is done or user exits.
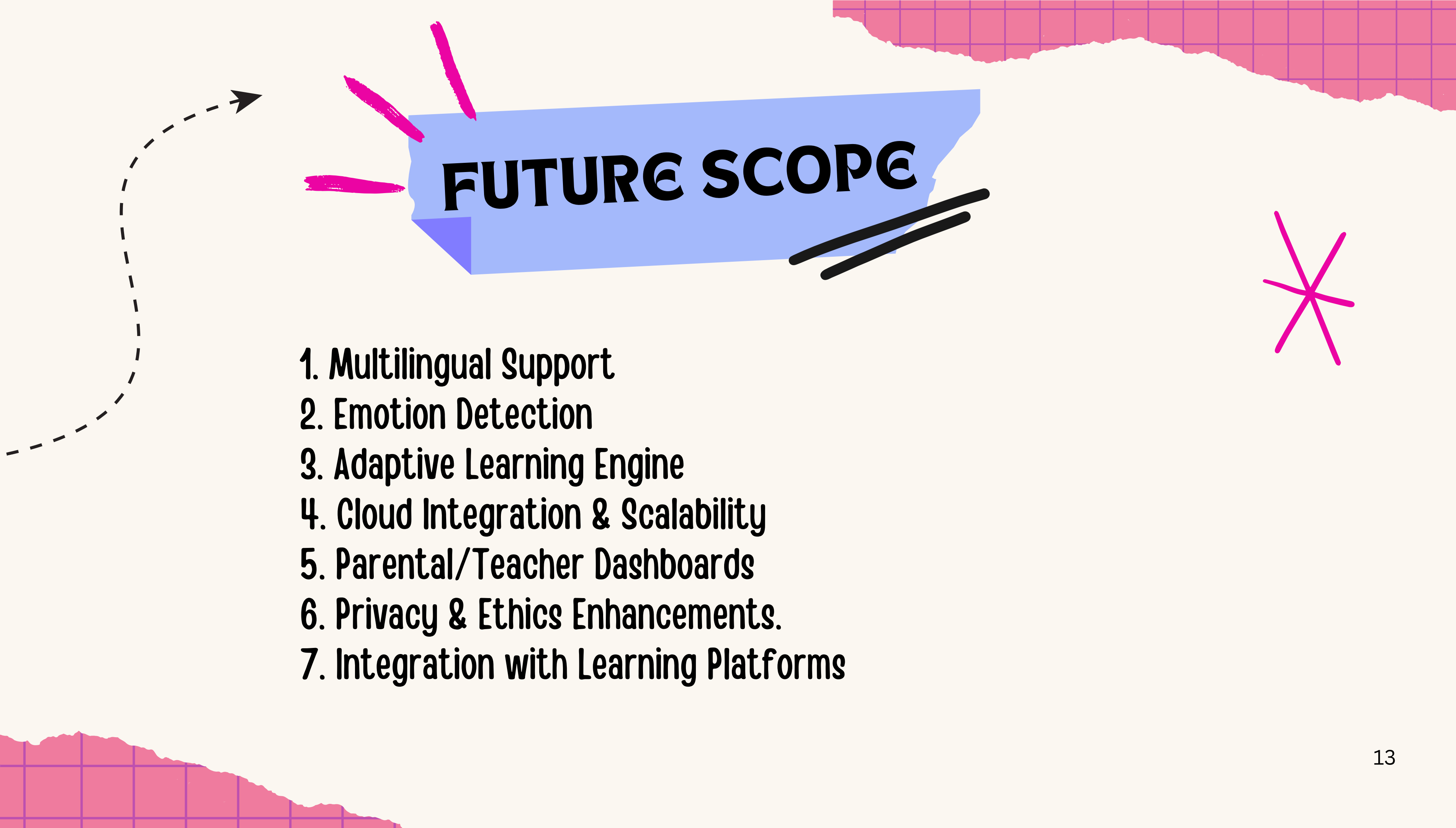- Program exits gracefully with final logs.

RESULTS

- Real-time Distraction Detection
- Audio Feedback System
- Speech Transcription
- Lecture Summarization
- Keyword-Based Highlights
- Multithreaded Performance
- Graceful Termination

CONCLUSION

- Dual Functionality

- Real-Time Feedback

- Speech Transcription and Summarization

FUTURE SCOPE

1. Multilingual Support
2. Emotion Detection
3. Adaptive Learning Engine
4. Cloud Integration & Scalability
5. Parental/Teacher Dashboards
6. Privacy & Ethics Enhancements.
7. Integration with Learning Platforms

THANK YOU!