Student Identification & Engagement Detection System - Requirements

# 1. Introduction

This system is designed to automatically identify students in a classroom using face recognition and evaluate their learning engagement levels. The goal is to distinguish between students who are actively engaged in learning and those who appear to be disengaged or forced into learning. The system generates analytical reports for administrators through a dashboard.

# 2. Objectives

- Identify students in real-time using face recognition.  
- Detect whether students are engaged or not engaged (forced).  
- Provide attendance and engagement statistics.  
- Generate visual reports and analytics for administrators.

# 3. Dataset Requirements

The system requires datasets for both student identification and engagement detection:

* a) Face Recognition Dataset
* - LFW (Labeled Faces in the Wild)
* - VGGFace2
* - Custom classroom dataset with multiple student face images
* b) Engagement / Attention Dataset
* - DAiSEE (Student Engagement Dataset with labels: boredom, engagement, frustration, confusion)
* - FER-2013 (Facial Expression Recognition dataset)
* - EmotiW (Emotion Recognition in the Wild, used for classroom engagement tasks)
* - Custom classroom video clips labeled as 'Engaged' or 'Not Engaged'

# 4. System Components

- Data Capture: Camera setup in classroom.  
- Face Detection & Recognition: OpenCV, DeepFace, FaceNet.  
- Engagement Detection: CNN/Transformer models trained on DAiSEE/FER2013.  
- Analytics & Storage: Database (PostgreSQL/MongoDB).  
- Dashboard: Django + React/Dash/Streamlit with graphs and charts.

# 5. Dashboard & Reporting Features

- Total students present.  
- Engaged vs Not Engaged statistics.  
- Individual student engagement history.  
- Graphs (Pie chart, Bar chart, Line chart, Heatmap).  
- Exportable reports for admin.

# 6. Model Training Flow

1. Pretrain on FER2013 and DAiSEE datasets.  
2. Fine-tune using classroom-specific student dataset.  
3. Train classifier to detect 'Engaged' vs 'Forced'.  
4. Deploy into real-time system connected with camera and dashboard.

# 7. Recommended Tech Stack

- Backend: Python (Flask/Django)  
- Machine Learning: TensorFlow / PyTorch  
- Face Recognition: DeepFace, FaceNet  
- Engagement Detection: CNN, RNN, Transformers  
- Dashboard: React.js or Streamlit/Dash  
- Database: PostgreSQL or MongoDB  
- Deployment: Docker + Cloud (AWS/GCP)