# FACE MASK DETECTION

# CSM D2 TEAM-08

## TEAM DETAILS

GALLA DURGA PRASAD (TL)  
TIRUMALASETTI SAI TEJA  
CHINTHALAPUDI MAHESH  
VAKADA SATYANARAYANA SAI  
PANCHADI RAMAKRISHNA

## Introduction

Face Mask Detection is a computer vision application that identifies whether individuals are wearing face masks in real-time.   
It has become crucial in ensuring public health and safety, especially during pandemics.   
This project employs **Deep Learning** and **OpenCV** to develop an accurate and efficient face mask detection system.  
Using a **pre-trained deep learning mode**l, the system can classify faces as "Mask" or "No Mask" with high accuracy.

## Abstract

This project focuses on **Face Mask Detection** using Convolutional Neural Networks **(CNNs)** and **OpenCV**.   
The system captures real-time video feeds, processes the input using a trained model, and classifies faces into two categories:   
**With Mask** and **Without Mask**.   
This project has significant applications in enforcing mask-wearing policies in public areas and improving safety measures.

## Technology

- **Python**: Core programming language used for development.  
- **OpenCV**: Handles image and video processing tasks.  
- **TensorFlow/Keras**: Provides deep learning models for face mask classification.  
- **CNN Model**: A pre-trained or custom-built convolutional neural network is used for classification.

## Uses and Applications

Face Mask Detection has a wide range of real-world applications, including:  
- **Public Safety**: Ensures compliance with mask-wearing regulations in public places.  
- **Healthcare Monitoring**: Helps in hospitals and clinics to enforce mask mandates.  
- **Smart Surveillance**: Enhances security camera systems with real-time mask detection alerts.  
- **Workplace Compliance**: Ensures employees adhere to mask guidelines in offices and industries.

## Steps to Build

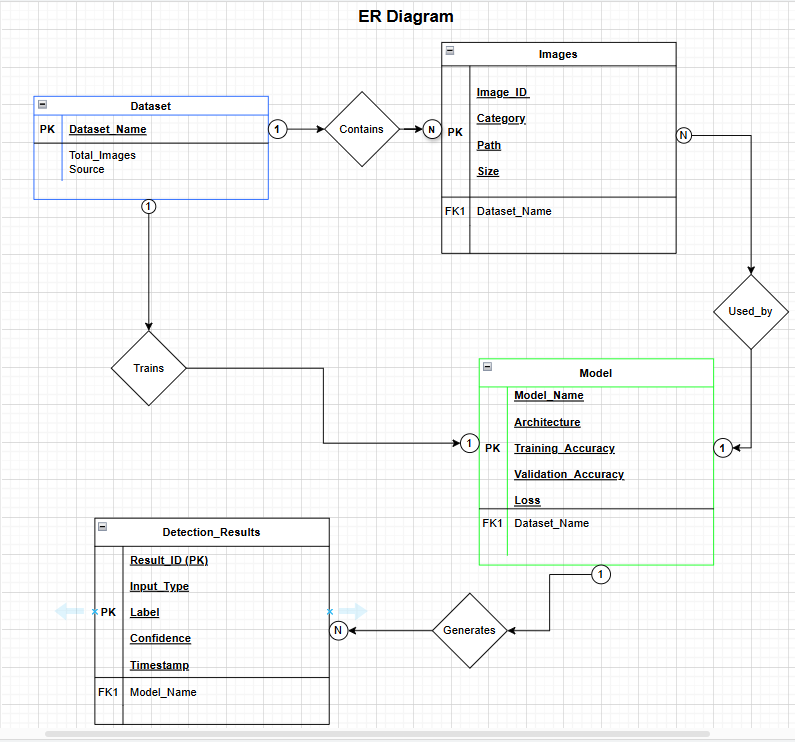
1. **Data Collection** : Use publicly available datasets of masked and unmasked faces.  
2. **Model Training** : Train a CNN model using TensorFlow/Keras with labeled face mask data.  
3. **Preprocessing** : Resize, normalize, and augment images to improve model accuracy.  
4. **Face Detection** : Use OpenCV’s pre-trained **Haar cascade** or **DNN face detector** to locate faces.  
5. **Mask Classification** : Apply the trained model to classify detected faces as "Mask" or "No Mask."  
6. **Real-time Detection** : Integrate with a live camera feed to detect masks in real-time.

## Work Flow

1. **Input**: The system captures video frames from a webcam or CCTV feed.  
2. **Processing**: Face detection is performed using OpenCV, followed by classification using a CNN model.  
3. **Output**: The system overlays "Mask" or "No Mask" labels on detected faces and provides alerts if needed.

**ER Diagram**

The Entity-Relationship (ER) Diagram illustrates the data flow and structure of the project:   
- Entities:   
- Dataset: Represents the Face Mask Dataset (Attributes: Dataset\_Name, Total\_Images, Source).   
- Images: Individual images within the dataset (Attributes: Image\_ID, Category, Path, Size).   
- Model: The trained detection model (Attributes: Model\_Name, Architecture, Training\_Accuracy, Validation\_Accuracy, Loss).   
- Detection\_Results: Output of the detection process (Attributes: Result\_ID, Input\_Type, Label, Confidence, Timestamp).   
- Relationships:   
- Dataset "Contains" Images (1-to-Many).   
- Dataset "Trains" Model (1-to-1).   
- Images "Used\_by" Model (Many-to-1).   
- Model "Generates" Detection\_Results (1-to-Many).   
The ER Diagram was created using Draw.io and saved as "ER Diagram.png" for reference.



## Conclusion

This project presents an effective **Face Mask Detection** system that utilizes deep learning and computer vision techniques.By leveraging **OpenCV** and **CNN models**, the system can efficiently classify masked and unmasked faces in real-time.   
This technology can be widely implemented for **public safety, healthcare, and surveillance**, ensuring better enforcement of mask-wearing policies.