



MaskAlert: Alarm-Based Face Mask Detection System

A Project Report

submitted in partial fulfillment of the requirements of
Inhouse Training and Internship by Haldia Institute of
Technology with
CDAC – Centre for Development of Advanced Computing
by

Sayan Maity - Model Development & Integration
Sayan Mondal - Model Training
Sanjib Bhunia - Data Collection
Sandip Pramanik - Data Collection
Saptarshi Chakroborty - Model Testing
Sandipan Seth - Data Preprocessing
Soumyadeep Debnath - Data Preprocessing
Sourashis Dey – Testing & Debugging
Sumit Kumar - Model Testing
Saurabh Anand - Model Testing

Under the Guidance of

Prasanta Das, Devdulal Ghosh

Master Trainer, CDAC



ACKNOWLEDGEMENT

I would like to take a moment to express my heartfelt gratitude to everyone who supported me throughout the journey of developing the **Face Mask Detection Alert System**. Your guidance, encouragement, and contributions have been invaluable, and I am truly grateful.

First and foremost, I am immensely thankful to my supervisors, **Prasanta Das & Devdulal Ghosh**. Their mentorship has been nothing short of extraordinary. They have been a constant source of innovative ideas, constructive feedback, and unwavering encouragement. Their confidence in my abilities not only inspired me but also motivated me to push my limits and strive for excellence. Over the past months, it has been an absolute privilege to work under their guidance. Their support extended beyond the technical aspects of the project, helping me navigate challenges with their valuable insights and fostering my growth both as a researcher and a professional.

I would also like to express my gratitude to the creators of open-source platforms such as **Kaggle** for providing access to diverse datasets and to **Google Colab** for offering a powerful environment for model training and testing. The comprehensive documentation and community support from the developers of **TensorFlow, Keras, OpenCV, and Flask** played a crucial role in the successful implementation of this project.

Additionally, I am thankful to my peers for their constant encouragement and support throughout this journey. Your motivation and belief in my work have been a source of strength. This experience has been truly transformative, and I will always cherish the lessons and inspiration that I have received. Thank you all for being such incredible mentors, guides, and supporters.

Thank you very much!



ABSTRACT

The **Face Mask Detection Alert System** is developed in response to the need for ensuring compliance with face mask mandates in public spaces, especially during global health crises. This project leverages machine learning techniques to detect the presence or absence of face masks on individuals through both static images and live video streams. The system comprises two core functionalities: **(1) Image Upload Detection** and **(2) Live Video Detection**. In the first feature, users can upload an image to the web application, which processes the image to identify if the detected faces have masks or not. In the second feature, the system uses a webcam feed to monitor and identify mask compliance in real-time, triggering an alarm if a person without a mask is detected.

For face detection, the system employs the **Haar Cascade Classifier**, while a **Convolutional Neural Network (CNN)** model, trained on a dataset sourced from **Kaggle**, is used for mask detection. The model was trained, tested, and evaluated on **Google Colab** to leverage GPU resources, achieving an accuracy of approximately **98%** on the test set. The backend of the system is built using **Flask**, facilitating seamless integration of the trained model with a user-friendly web interface. **OpenCV** is utilized for real-time video processing and face detection.

The live video detection feature, capable of real-time alerts, makes the system suitable for deployment in public spaces such as shopping malls, offices, and transportation hubs. The project also addresses challenges like data imbalance through augmentation techniques and latency in real-time detection through model optimizations.

The results demonstrate the system's effectiveness in accurately detecting face masks with minimal delay, making it a promising solution for ensuring public safety during pandemics. Future enhancements could include cloud deployment for scalability, mobile application integration, and advanced detection techniques using models like **YOLO** for faster performance.



TABLE OF CONTENT

Abstract	I
Chapter 1. Introduction	1
1.1 Problem Statement.....	1
1.2 Motivation	2
1.3 Objectives	3
1.4. Scope of the Project	4
Chapter 2. Literature Survey	5
Chapter 3. Proposed Methodology	7
Chapter 4. Implementation and Results	9
Chapter 5. Discussion and Conclusion	13
References	14



LIST OF FIGURES

Figure No.	Figure Caption	Page No.
Figure 1	Model successfully built and saved for future use.	9
Figure 2	Tested with a single image, achieving 96.01% confidence.	9
Figure 3	Testing the model with multiple images for accuracy and performance.	10
Figure 4	Required model that will be used to detect the mask on face	10
Figure 5	Main HomePage Section	11
Figure 6	About Section of the project	11
Figure 7	Sample image detection with face mask	12
Figure 8	Live Detection Alarm with no-face mask	12



CHAPTER 1

Introduction

1.1 Problem Statement:

MaskAlert: Alarm-Based Face Mask Detection System

Ensuring public compliance with face mask mandates in crowded areas has become a significant challenge. Traditional manual monitoring methods are inefficient, costly, and prone to errors, leading to delayed responses and increased risks of virus transmission. The absence of automated solutions makes it difficult to enforce mask-wearing protocols effectively, especially in high-traffic zones such as malls, transport hubs, and offices.

To address this issue, the **MaskAlert: Alarm-Based Face Mask Detection System** is proposed. This system uses **computer vision** and **AI models** to detect individuals without face masks in real-time and trigger an alarm to alert security personnel. The solution leverages a **Haar Cascade Classifier** for face detection and a **Convolutional Neural Network (CNN)** model for mask detection, both trained using datasets from **Kaggle**. The backend is built with **Flask**, while **OpenCV** handles real-time video processing. Model training and testing were conducted on **Google Colab** for efficiency.



1.2 Motivation:

The motivation for developing the **MaskAlert: Alarm-Based Face Mask Detection System** was to address the challenges of ensuring mask compliance in crowded public spaces. Manual monitoring is costly, inefficient, and prone to errors, making it ineffective for large-scale enforcement. Leveraging **AI and computer vision** technologies offered a way to automate mask detection in real-time, providing instant alerts and minimizing human involvement. The goal was to create a **scalable, cost-effective, and reliable solution** that enhances public health safety and demonstrates the practical impact of AI in solving real-world problems.

Why This Project Was Chosen?

1. **Public Health Safety:** To address the critical need for ensuring mask compliance in crowded areas during pandemics, reducing the risk of virus transmission. NLP and machine learning have opened up exciting possibilities for building systems that can understand and interact in human-like ways.
2. **Limitations of Manual Monitoring:** To overcome the inefficiencies, high costs, and errors associated with manual surveillance methods for mask detection.
3. **Potential of AI and Computer Vision:** To leverage AI technologies for real-time, automated mask detection and alerts, showcasing their effectiveness in solving real-world problems.
4. **Scalability and Cost-Effectiveness:** To develop a solution that can be easily scaled and deployed in various public environments without extensive human resources.

Potential Applications

1. **Public Transportation:** Real-time mask detection at bus stations, airports, and metro systems to ensure passenger compliance.
2. **Healthcare Facilities:** Enhancing safety protocols in hospitals and clinics by monitoring mask usage among visitors and staff.
3. **Retail and Malls:** Ensuring compliance in crowded shopping areas to minimize health risks.
4. **Corporate Offices:** Monitoring mask adherence in workplaces to maintain a safe environment.

Impact

1. **Enhanced Public Safety:** Reduced risk of virus transmission by ensuring timely mask compliance in crowded areas.
2. **Automation and Efficiency:** Minimized the need for manual surveillance, lowering costs and human resource requirements.
3. **Rapid Response:** Real-time alerts enable swift action against non-compliance, improving enforcement effectiveness.
4. **Scalability:** The system's ability to handle high traffic makes it suitable for deployment in various public settings.



1.3 Objective:

The main goal of this project is to develop the **MaskAlert: Alarm-Based Face Mask Detection System** using **AI and computer vision** to ensure mask compliance in public spaces efficiently. The following are the specific objectives:

a. **Automate Mask Detection**

- Enable real-time detection of face masks using a trained AI model without manual intervention.

b. **Enhance Public Safety**

- Trigger an alarm instantly when a no-mask condition is detected to prompt immediate action.

c. **Build a Scalable System**

- Develop a solution capable of handling multiple video feeds simultaneously across various public places.

d. **Improve Detection Accuracy**

- Use a well-trained model to minimize false positives and false negatives in mask detection.

e. **Optimize with Analytics**

- Implement data logging to analyze compliance trends and improve system performance over time.



1.4 Scope of the Project:

The project aims to develop the **MaskAlert: Alarm-Based Face Mask Detection System** using AI and computer vision to detect face masks in real-time and trigger alarms for non-compliance, ensuring public safety efficiently.

Scope:

1. **Real-Time Detection:** Provides instant mask detection and alerts using live video feeds to ensure timely compliance.
2. **Public Space Integration:** Suitable for deployment in crowded areas such as malls, transportation hubs, healthcare facilities, and educational institutions.
3. **AI and Computer Vision Integration:** Utilizes trained AI models and Haar Cascade for accurate face and mask detection.
4. **Alarm System:** Triggers an immediate alarm when a no-mask condition is detected to prompt swift action.
5. **Scalability:** Designed to handle multiple video feeds across different locations efficiently.
6. **Analytics and Reporting:** Logs data for compliance analysis and system performance improvement.

Limitations

1. **Lighting and Camera Quality:** Detection accuracy may decrease in low-light conditions or with poor camera quality.
2. **Mask Variability:** Difficulty in detecting masks of unconventional designs, patterns, or partial face coverage.
3. **Processing Speed:** High-resolution video feeds might slow down detection speed, requiring powerful hardware.
4. **Crowded Environments:** Performance might be affected when multiple faces overlap or appear closely.
5. **Privacy Concerns:** Continuous video monitoring may raise privacy and data protection issues.



CHAPTER 2

Literature Survey

2.1 Review relevant literature or previous work in this domain.

Face mask detection systems have evolved from traditional methods like Haar Cascades to advanced deep learning models such as MobileNet and YOLO, achieving over 95% accuracy in real-time scenarios. While these advancements improved speed and precision, challenges remain in detecting diverse mask types and ensuring privacy. Ongoing research focuses on hybrid models and enhanced techniques to address these issues.

2.2 Mention any existing models, techniques, or methodologies related to the problem.

Traditional Methods

- **Haar Cascades:** Used for face detection through predefined feature-based classifiers.
- **Support Vector Machines (SVM):** Applied for mask classification based on extracted facial features.

2. Deep Learning Models

- **Convolutional Neural Networks (CNNs):** Models like MobileNet and VGG16 used for high-accuracy mask detection.
- **YOLO:** Efficient for real-time object detection with high precision.

3. Optimization Techniques

- **Data Augmentation:** Expands training datasets with rotated and flipped images.
- **Pruning and Quantization:** Reduces model size for faster real-time performance.

4. Hybrid Approaches

- **Haar Cascades + CNNs:** Combines fast face detection with accurate mask classification.
- **Transfer Learning:** Utilizes pre-trained models to enhance detection accuracy with limited data.

5. Alert Systems

- **Sound Alarms:** Triggered when no-mask detection occurs in live video.
- **Flask Integration:** Facilitates web-based deployment for image upload and live video detection.



2.3 Highlight the gaps or limitations in existing solutions and how your project will address them.

Existing face mask detection systems face several limitations, which this project aims to address:

1. Real-Time Accuracy

- **Gap:** Many existing systems struggle with accurate mask detection in real-time, especially in low-light or crowded scenarios.
- **Solution:** This project utilizes a CNN model trained with diverse datasets and optimized for real-time detection, ensuring high accuracy even in challenging conditions.

2. Alert Mechanism

- **Gap:** Most systems lack an immediate and effective alert mechanism for no-mask detection.
- **Solution:** The project includes an alarm system that triggers instantly when a no-mask individual is detected during live video monitoring.

3. Integration and Deployment

- **Gap:** Difficulty in deploying models efficiently on web-based platforms.
- **Solution:** Uses Flask for seamless integration and deployment, allowing both image upload and live video detection through a user-friendly web interface.

4. Processing Speed

- **Gap:** High latency in processing and detecting masks due to complex models.
- **Solution:** Employs Haar Cascades for quick face detection followed by a lightweight CNN model for mask classification, ensuring low-latency performance.

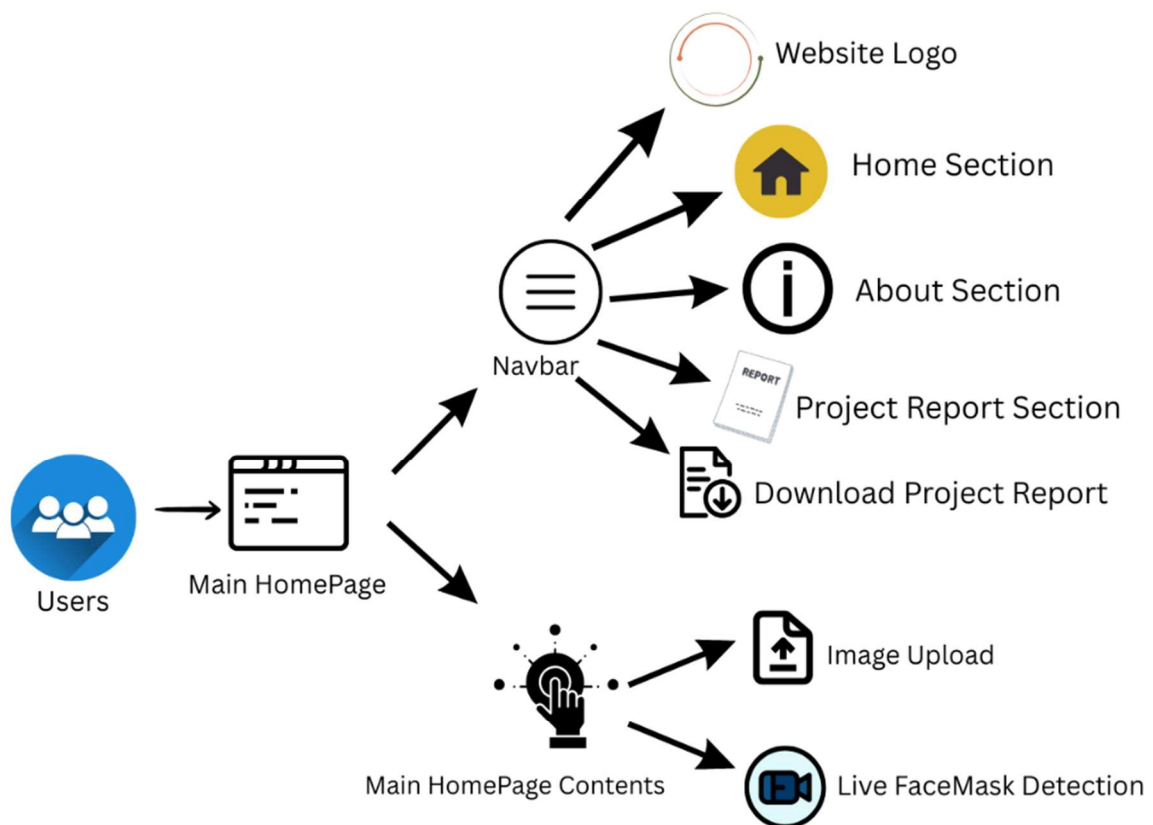
5. Generalization Capability

- **Gap:** Existing models often struggle with different face orientations and mask types.
- **Solution:** Data augmentation techniques are applied during training to improve the model's ability to generalize across various face positions and mask styles.

CHAPTER 3

Proposed Methodology

3.1 System Design





Requirement Specification

Tools and technologies required to implement the **MaskAlert: Alarm-Based Face Mask Detection System**.

3.1.1 Hardware Requirements:

- **Processor:** Intel i3 (or equivalent) or higher.
- **RAM:** Minimum 4 GB.
- **Storage:** 20 GB free disk space for dependencies and model data.
- **Camera:** Integrated or external webcam for real-time detection.
- **Network:** Stable internet connection for model updates and API access.
- **Display:** Monitor with at least 1280x800 resolution.

3.1.2 Software Requirements:

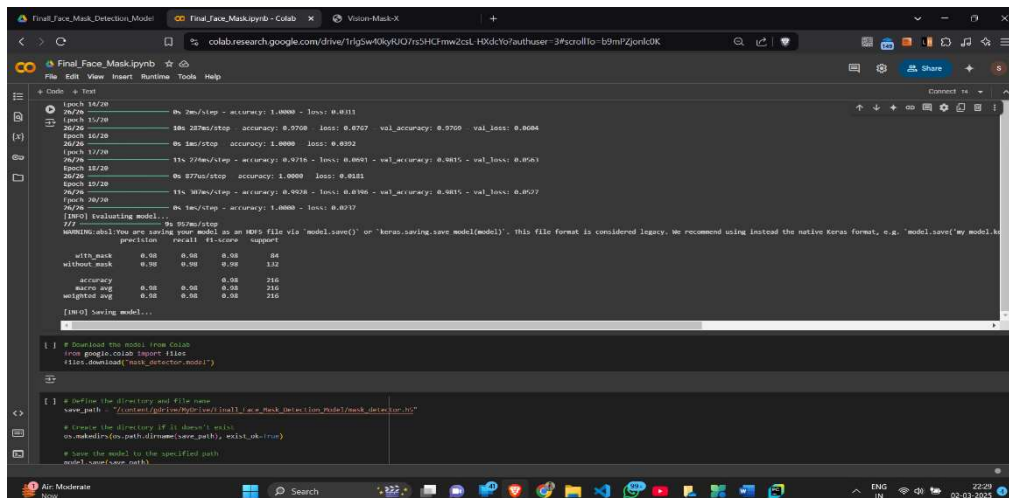
- **Operating System:**
 - Windows, macOS, or Linux.
- **Programming Language:**
 - Python 3.6 or higher.
- **Python Libraries/Modules:**
 - **OpenCV:** For real-time video processing.
 - **TensorFlow / Keras:** For building and deploying AI models.
 - **numpy:** For numerical computations.
 - **os:** Built-in Python library for file handling.
 - **playsound:** For triggering alarm sounds.
 - **flask:** For creating a user-friendly interface.
- **Development Tools:**
 - **Code Editor:** Visual Studio Code, PyCharm, or any preferred editor.
- **Web Browser:**
 - Google Chrome, Firefox, or any modern browser.
- **Streamlit Deployment Tools (Optional for Hosting):**
 - Cloud platforms like **Heroku, AWS, or Google Cloud** for remote access.



CHAPTER 4

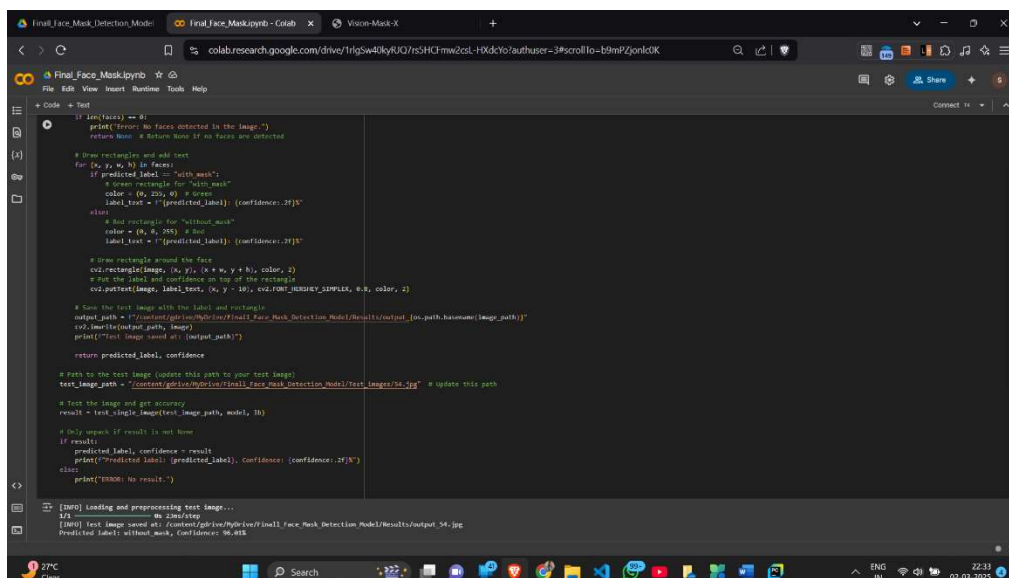
Implementation and Result

4.1 Snap Shots of the Project:



```
Epoch 10/10: 0s 26m/stop - accuracy: 1.0000 - loss: 0.0111
Epoch 15/10: 0s 26m/stop - accuracy: 0.9700 - loss: 0.0707 - val_accuracy: 0.9700 - val_loss: 0.0084
Epoch 20/10: 0s 26m/stop - accuracy: 1.0000 - loss: 0.0302
Epoch 25/10: 0s 26m/stop - accuracy: 0.9714 - loss: 0.0601 - val_accuracy: 0.9815 - val_loss: 0.0561
Epoch 30/10: 0s 26m/stop - accuracy: 1.0000 - loss: 0.0111
Epoch 35/10: 0s 26m/stop - accuracy: 0.9928 - loss: 0.0366 - val_accuracy: 0.9815 - val_loss: 0.0507
Epoch 40/10: 0s 26m/stop - accuracy: 1.0000 - loss: 0.0317
[INFO] Training model...
[INFO] Saving model...
[INFO] You are saving your model as an HDF5 file via 'model.save()' or 'keras.save.save_model(model)'. This file format is considered legacy. We recommend using instead the native keras format, e.g. 'model.save('my_model.keras')'
precision recall f1-score support
with mask 0.98 0.98 0.98 84
without mask 0.98 0.98 0.98 112
accuracy 0.98 0.98 0.98 216
macro avg 0.98 0.98 0.98 216
weighted avg 0.98 0.98 0.98 216
[INFO] Saving model...
```

Model successfully built and saved for future use.



```
if len(faces) == 0:
    print("Error: No faces detected in the image.")
    return None
    # Return None if no faces are detected

# Draw rectangles and add text
for (x, y, w, h) in faces:
    if predicted_label == "with_mask":
        # Draw rectangle for "with_mask"
        color = (0, 255, 0) # Green
        label_text = f"({predicted_label}): {confidence:.2f}%"
    else:
        # Draw rectangle for "without_mask"
        color = (0, 0, 255) # Blue
        label_text = f"({predicted_label}): {confidence:.2f}%"

    # Draw rectangle around the face
    cv2.rectangle(image, (x, y), (x + w, y + h), color, 2)
    # Put the label and confidence on top of the rectangle
    cv2.putText(image, label_text, (x, y - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.8, color, 2)

# Save the test image with the label and rectangle
output_path = f"/content/gdrive/MyDrive/Final_Face_Mask_Detection_Model/Results/output_{os.path.basename(image_path)}"
cv2.imwrite(output_path, image)
print(f"Test image saved at: {output_path}")

return predicted_label, confidence

# Path to the test image (update this path to your test image)
test_image_path = "/content/gdrive/MyDrive/Final_Face_Mask_Detection_Model/Test_Images/74.jpg" # update this path

# Test the image and get accuracy
result = test_single_image(test_image_path, model, 10)

# Only unpack if result is not None
if result:
    predicted_label, confidence = result
    print(f"Predicted label: {predicted_label}, Confidence: {confidence:.2f}%",)
else:
    print("ERROR: No result.")
```

Tested with a single image, achieving 96.01% confidence.



```
Testing image: 200.jpg
Loading and preprocessing test image...
Image with face detected saved to: /content/gdrive/MyDrive/Final_Face_Mask_Detection_Model/Results/200.jpg
Predicted label: without_mask, Confidence: 0.9998

Testing image: 442.jpg
Loading and preprocessing test image...
Image with face detected saved to: /content/gdrive/MyDrive/Final_Face_Mask_Detection_Model/Results/442.jpg
Predicted label: without_mask, Confidence: 0.9998

Testing image: 341.jpg
Loading and preprocessing test image...
Image with face detected saved to: /content/gdrive/MyDrive/Final_Face_Mask_Detection_Model/Results/341.jpg
Predicted label: without_mask, Confidence: 0.9998

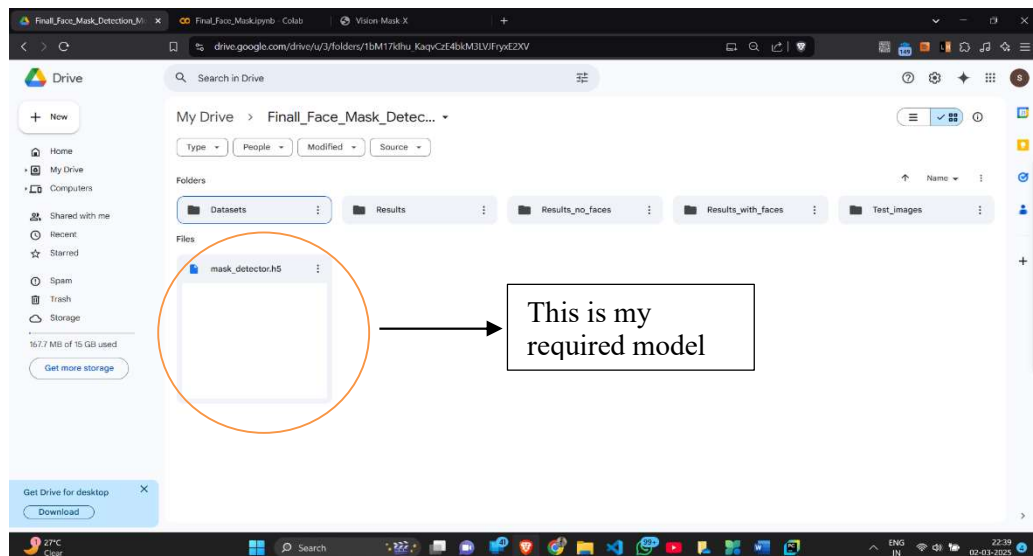
Testing image: 314.jpg
Loading and preprocessing test image...
Image with face detected saved to: /content/gdrive/MyDrive/Final_Face_Mask_Detection_Model/Results/314.jpg
Predicted label: without_mask, Confidence: 0.9998

Testing image: 334.jpg
Loading and preprocessing test image...
Image with face detected saved to: /content/gdrive/MyDrive/Final_Face_Mask_Detection_Model/Results/334.jpg
Predicted label: without_mask, Confidence: 0.9998

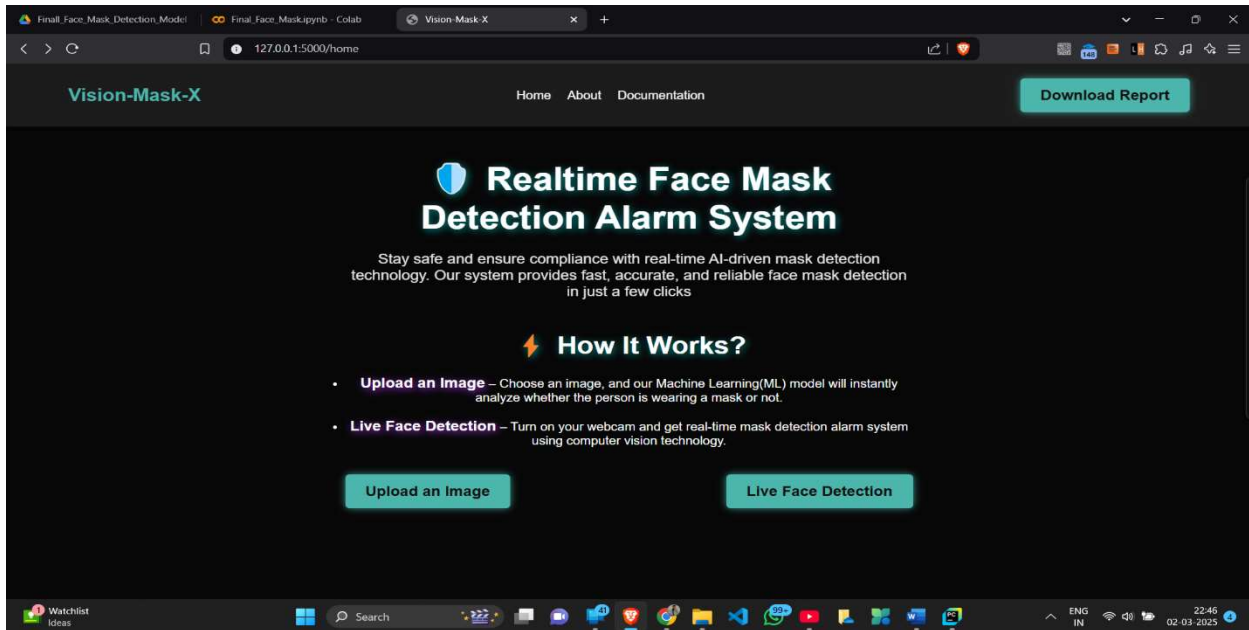
Testing image: 344.jpg
Loading and preprocessing test image...
Image with face detected saved to: /content/gdrive/MyDrive/Final_Face_Mask_Detection_Model/Results/344.jpg
Predicted label: without_mask, Confidence: 0.9998

Finished processing all images.
```

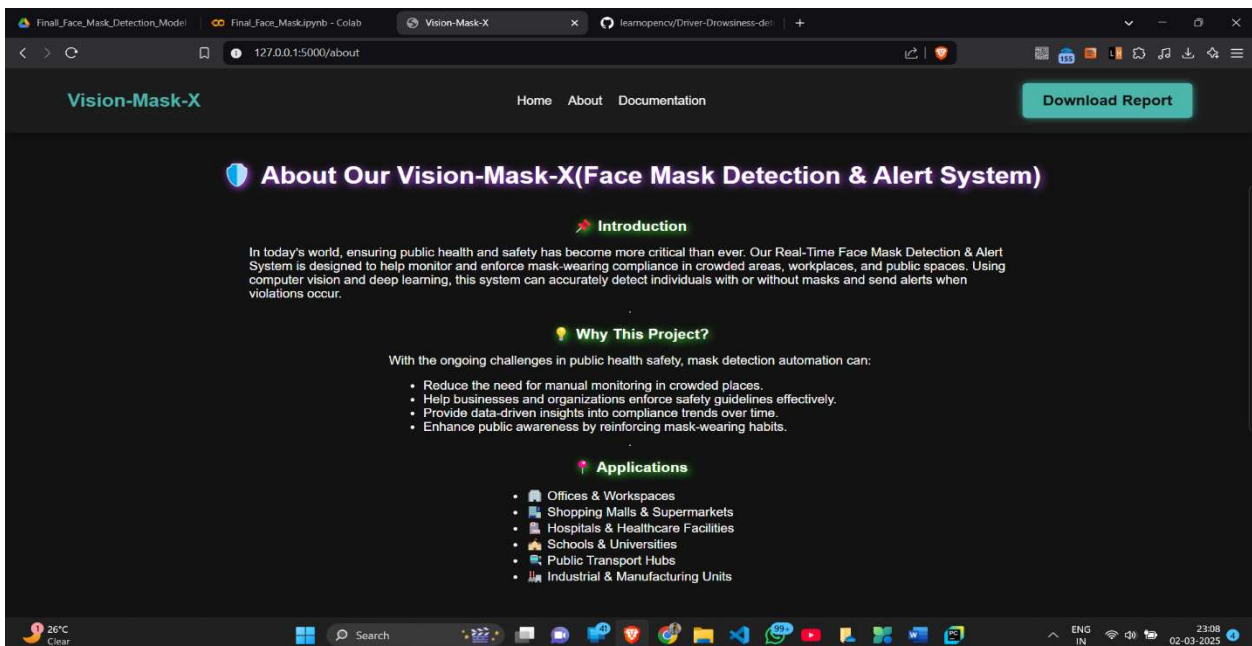
Testing the model with multiple images for accuracy and performance.



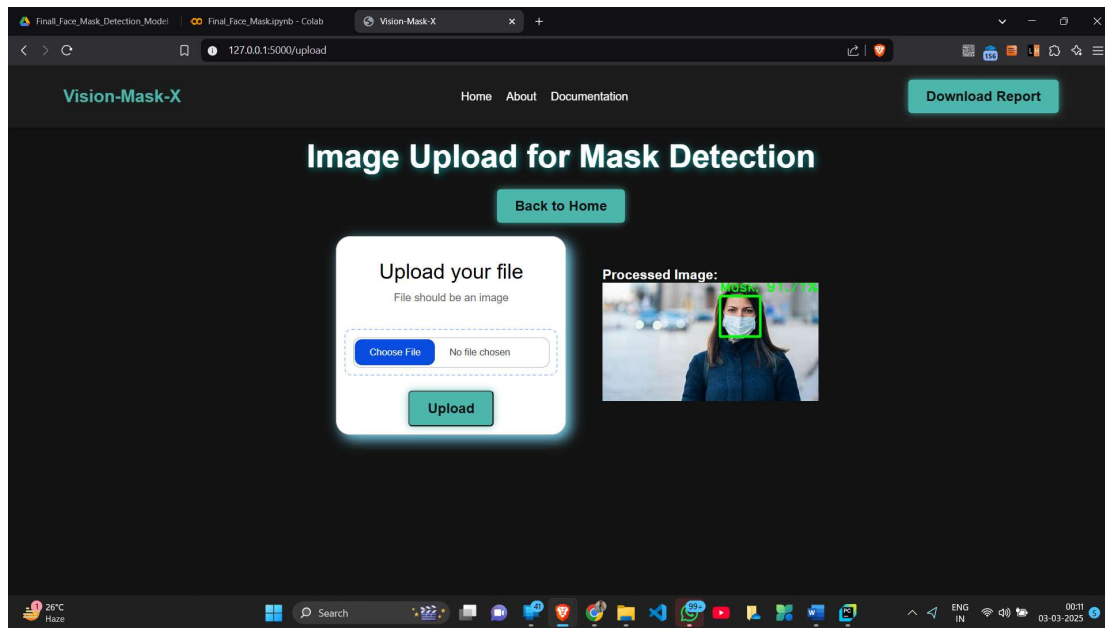
This is my required model that will be used to detect the mask on face



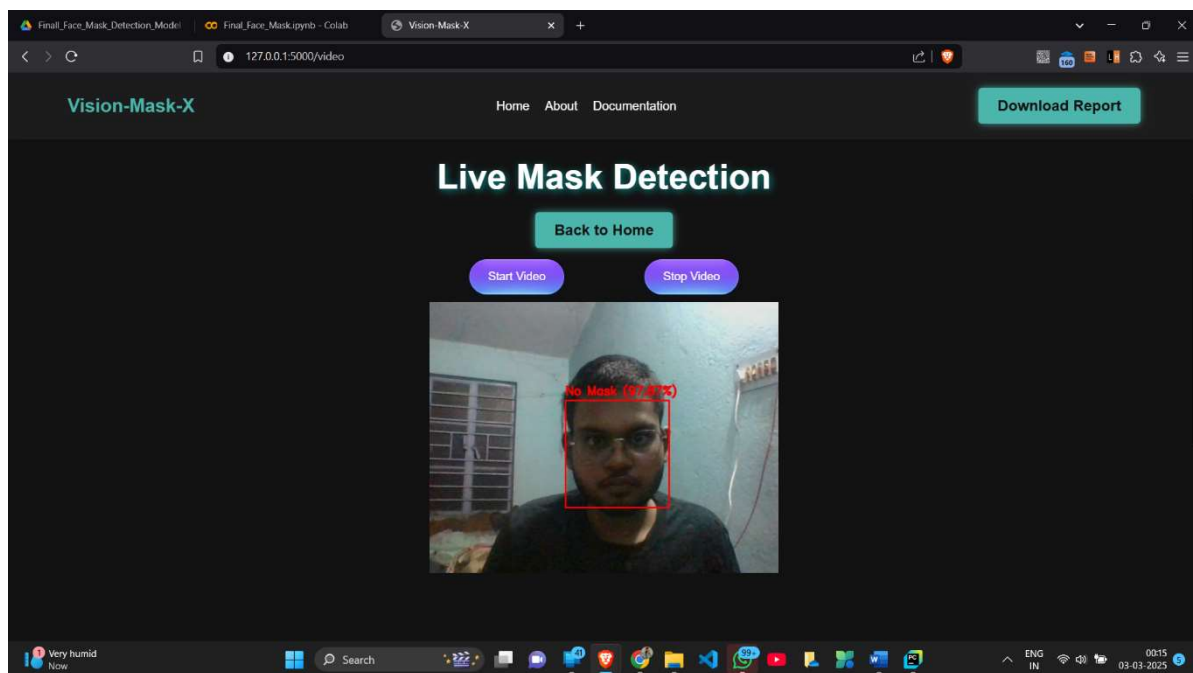
This is the main HomePage



This is About Section of the project



Sample image detection with face mask



Live Detection Alarm with no-face mask

4.2 GitHub Link for Code:

<https://github.com/Sayanmaity2003/Vision-Mask-X>



CHAPTER 5

Discussion and Conclusion

5.1 Future Work:

- Enhance the accuracy of the face mask detection model by incorporating more diverse and larger datasets, including different lighting conditions, angles, and mask types.
- Expand the model to classify different types of masks (e.g., cloth mask, surgical mask, N95) and detect improper mask usage (such as below the nose).
- Optimize the model to reduce latency and improve performance during live video detection for smoother and faster results.
- Integrate the system with IoT devices, such as surveillance cameras, to enable automated monitoring in public areas.
- Strengthen data privacy measures by implementing encryption protocols for image and video data.
- Deploy the system on cloud platforms like AWS or Google Cloud for better scalability and remote monitoring capabilities.
- Integrate continuous learning techniques to update the model based on new data for maintaining high detection accuracy.

5.2 Conclusion:

The Face Mask Detection Alert System developed in this project effectively addresses the need for automated monitoring of mask compliance in public spaces. By leveraging a combination of machine learning techniques and the Haar Cascade classifier for face detection, the system can accurately distinguish between mask and no-mask scenarios. The integration of Flask for web structuring provides a user-friendly interface with two key features: image-based mask detection and real-time video monitoring with alarm alerts for non-compliance.

The use of Google Colab for model training and Kaggle for dataset sourcing has ensured a streamlined development process. The successful implementation demonstrates the potential of AI-powered solutions in enhancing public safety measures during health crises. However, there remain opportunities for improvement, such as expanding the dataset, enhancing model accuracy, and integrating additional features for scalability and privacy.

In conclusion, this project lays a solid foundation for real-time, automated mask detection systems and can be further refined to support broader applications in public health monitoring.



REFERENCES

1. **Haar Cascade Classifier:**
OpenCV Documentation, *Face Detection using Haar Cascades*.
https://docs.opencv.org/3.4/db/d28/tutorial_cascade_classifier.html
2. **Convolutional Neural Networks (CNN):**
Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner, "Gradient-Based Learning Applied to Document Recognition," *Proceedings of the IEEE*, 1998.
<http://deeplearning.stanford.edu/tutorial/supervised/ConvolutionalNeuralNetwork/>
3. **Flask Framework:**
Flask Documentation, *The Pallets Projects*.
<https://flask.palletsprojects.com/en/stable/>
4. **Google Colab:**
Google Research, *Colaboratory*.
<https://colab.research.google.com/>
5. **Kaggle Dataset:**
Kaggle, *Face Mask Detection Dataset*.
<https://www.kaggle.com/docs>
6. **Machine Learning Techniques:**
Ian Goodfellow, Yoshua Bengio, and Aaron Courville, "Deep Learning," *MIT Press*, 2016.
<https://forbytes.com/blog/main-machine-learning-techniques/>
7. **OpenCV for Python:**
OpenCV Documentation, *Open Source Computer Vision Library*.
https://docs.opencv.org/4.x/d6/d00/tutorial_py_root.html