# ABSTRACT

This project introduces an innovative real-time cigarette smoking detection system, responding to the global health hazard posed by smoking. The system is meticulously designed to prevent and mitigate the adverse effects of cigarette smoking by employing state-of-the-art deep learning models for rapid detection. In this paper, we present a comprehensive approach to real-time cigarette detection, harnessing the capabilities of TensorFlow Lite. Our system aims to demonstrate its efficacy in live video streams, enabling immediate intervention to address smoking instances. A key component of the system is its integration of an alert mechanism, which promptly notifies relevant authorities upon detecting smoking activities, allowing for swift and appropriate actions. The underlying deep learning model is specifically trained for cigarette detection using a diverse dataset sourced from Roboflow. This dataset comprises images encompassing both cigarettes and non-cigarettes, ensuring the model's robustness and adaptability to real-world scenarios. The training process incorporates essential Python modules, including os, argparse, cv2, numpy, sys, time, threading, and importlib, to ensure the development of a robust and efficient model. Operating on the latest version and leveraging GPU capabilities for optimal performance, the system is designed to minimize false positives, thereby enhancing its accuracy in real-world scenarios. The user interface, developed in Python, plays a crucial role in enhancing user interaction. It allows for the easy configuration of detection settings, live monitoring, and real-time alert reception, ensuring a user-friendly and accessible experience. This project stands as a proactive solution, addressing public health and safety concerns by enforcing no-smoking policies. The integration of TensorFlow Lite's capabilities with a user-friendly interface and real-time alerting positions the system as an effective tool for immediate interventions in areas where smoking is prohibited. In our evaluation, the system demonstrated a high accuracy rate of 79.8% in detecting cigarettes across various real-world scenarios. This performance consistency was observed under different lighting conditions, distances, and angles, showcasing the system's reliability in diverse environments. Additionally, the real-time processing capability was validated, revealing an average processing time of 0.3 seconds per frame. This affirms the system's efficiency and responsiveness, making it a reliable tool for swift and accurate smoking detection.This research project contributes to the advancement of real-time cigarette smoking detection systems. The integration of cutting-edge technology, a diverse training dataset, and a user-friendly interface positions the system as a valuable asset in promoting public health and safety by enforcing no-smoking policies.

# INDEX

# 01.INTRODUCTION

# CHAPTER 1: INTRODUCTION

## 1.1 INTRODUCTION:

Cigarette smoking stands as a significant global health concern, constituting a leading cause of preventable deaths. According to the World Health Organization (WHO), smoking contributes to over 8 million deaths annually, with more than 7 million directly linked to tobacco use. Second-hand smoke exposure further results in over 1 million deaths each year. The associated risks of chronic diseases, including lung cancer, cardiovascular disease, and chronic obstructive pulmonary disease (COPD), emphasize the critical need to address this public health issue. Additionally, non-fatal health conditions such as respiratory infections, poor oral health, and compromised reproductive health compound the far-reaching consequences of cigarette smoking. Given these substantial health hazards, real-time detection of smoking instances emerges as a crucial strategy to prevent and mitigate the harmful effects of this pervasive habit.

Recent advancements in deep learning models have shown notable potential across various computer vision applications, particularly in object detection. Object detection, involving the identification and localization of objects within images or video feeds, plays a pivotal role in enhancing surveillance and intervention capabilities. In this research project, we veer away from traditional approaches and instead focus on training a custom model using TensorFlow Lite, a lightweight framework suitable for real-time applications. This custom model is specifically designed to detect instances of cigarette smoking through a live camera feed.

The introduction of TensorFlow Lite underscores our commitment to efficiency and responsiveness in addressing the urgent need for smoking detection and intervention. The research paper aims to delve into the architecture, training methodologies, and potential impact of the proposed system on public health and safety.

By adopting a novel approach, this research project seeks to contribute to the ongoing efforts to combat smoking-related health hazards. With a comprehensive exploration of the risks associated with smoking and a focus on real-time detection leveraging cutting-edge technology, the project aspires to offer an effective tool for immediate interventions. The subsequent sections of this paper will delve into the intricate details of the custom model's development, training processes, and the system's anticipated impact on public health outcomes.

## 1.2 NEED FOR STUDY:

The imperative to conduct this study on real-time cigarette smoking detection using TensorFlow Lite arises from various critical factors, emphasizing the pressing need to address the widespread health risks associated with smoking. The study seeks to tackle the enforcement challenges posed by no-smoking policies, leverage the transformative potential offered by deep learning technology, contribute significantly to broader public health initiatives, address ethical considerations pertaining to exposure, and foster essential interdisciplinary collaboration to develop effective and sustainable smoking detection solutions. In contemporary society, cigarette smoking stands as a formidable global health concern, contributing substantially to preventable deaths and a myriad of chronic diseases. The need for innovative and efficient solutions to detect smoking instances in real-time has become increasingly evident as traditional enforcement methods struggle to keep pace with evolving challenges. The enforcement of no-smoking policies in public spaces has encountered limitations primarily due to the reliance on manual monitoring. Automated systems have emerged as a necessary and practical solution to enhance compliance and overall public safety. By employing advanced technologies such as TensorFlow Lite, this study endeavors to bridge the gap between policy intent and effective enforcement, thereby creating smoke-free environments and reducing smoking-related diseases. The advancements in deep learning, particularly through TensorFlow Lite, present a promising avenue for the development of robust and efficient smoking detection mechanisms. TensorFlow Lite's lightweight framework is well-suited for real-time applications, ensuring swift analysis and intervention in smoke-free zones. The study aims to harness this technology to create a responsive system capable of prompt identification and deterrence of smoking instances. By aligning with broader public health initiatives, the study seeks to contribute actively to the creation of healthier environments. The reduction of smoking-related diseases is a crucial aspect of this endeavor, and the proposed smoking detection system, integrated with TensorFlow Lite, aligns with the overarching goal of promoting public health and safety. Ethical considerations play a pivotal role in shaping the trajectory of this study. The importance of protecting individuals, particularly vulnerable populations, from the harmful effects of secondhand smoke cannot be overstated. The societal responsibility to establish safer public spaces is reinforced by the ethical imperative to safeguard the health and well-being of all individuals. The smoking detection system, powered by TensorFlow Lite, embodies this commitment to ethical standards by providing a proactive means of minimizing exposure to secondhand smoke. Ultimately, the success of this study hinges on interdisciplinary collaboration. Technology experts, public health professionals, and policymakers must join forces to integrate technical innovation seamlessly with comprehensive public health strategies. This collaboration is imperative for advancing the goal of creating effective and sustainable smoking detection solutions that align with the evolving landscape of public health challenges.

## 1.3 LITERATURE REVIEW:

Smoke Detection Based on Deep Learning - Alibaba Cloud. These deep learning-based smoke detection systems can identify smoke in various environments, encompassing both indoor and outdoor settings. The systems exhibit versatility in detecting diverse types of smoke, including cigarette smoke, wood smoke, and more. Notably, they can operate effectively under different environmental conditions such as low light, high humidity, and various other challenging scenarios. The advantages of deep learning-based smoke detection systems extend beyond traditional smoke detectors, offering quicker and more accurate smoke detection capabilities.

Saurabh Singh Thakur, Pradeep Poddar & Ram Babu Roy assert that accurately detecting smoking activity amid the complexities of daily living, monitored by wearable devices, poses a significant challenge. Their study endeavors to develop a real-time machine learning-based modeling framework for identifying smoking activity amidst various daily activities.

Addressing forest fire and smoke detection, this research explores deep learning-based approaches without forgetting. The study delves into fire and smoke detection from images using AI-driven computer vision techniques. Convolutional Neural Networks (CNN), a subtype of Artificial Intelligence (AI), have demonstrated superior performance in image classification and other computer vision tasks. However, their extended training time can be a limiting factor. Additionally, pretrained CNNs may underperform when faced with insufficient datasets. To overcome this challenge, the research employs transfer learning on pretrained models.

Smoke Detection Based on Deep Convolutional Neural Networks to enhance smoke detection accuracy, a novel approach based on deep convolutional neural networks is proposed. This approach allows end-to-end training from raw pixel values to classifier outputs, enabling automatic feature extraction from images.

In recent years, the surge in smoking within no-smoking zones has driven the development of innovative surveillance systems, especially in smart cities. Ali Khan et al. introduce an AI-based system tailored for smart cities, addressing challenges through a robust smoker detection framework and a curated dataset. Their proposed solution achieves 96.87% accuracy, complementing Saurabh Singh Thakur et al.'s work on real-time smoking detection using wearable devices. Together, these studies advance AI-driven solutions for monitoring smoking activities in varied environments.

This paper by Waynebert Jan D. Cabanto et al. introduces a method for real-time multi-person smoking event detection. Utilizing Histogram of Oriented Gradients for human detection, centroid tracking, CNN for action recognition, and SVM for smoke detection, the proposed approach showcases potential effectiveness in creating smoke-free environments, contributing to the literature on automated smoking event detection.

In their contribution to ICT Analysis and Applications, Anushka Sharma, Trapti Mishra, Jyoti Kukade, and Aadya Golwalkar emphasize the transformative potential of AI in object detection. Their study explores TensorFlow's diverse deep learning models for real-time, high-accuracy object detection, particularly employing the Single Shot Multibox Detector (SSD) architecture. By showcasing the practical applications across autonomous driving, robotics, and surveillance, the authors highlight the evolving landscape of computer vision technologies.

In their study published in SSRG International Journal of VLSI & Signal Processing, Ms. S. Supraja and Dr. P. Ranjith Kumar present an Intelligent Traffic Signal Detection System utilizing SqueezeNet CNN. Addressing the challenges in recognizing numerous traffic-sign categories, the proposed framework showcases improvements, demonstrating enhanced overall performance with minimal error rates. Their work contributes to the evolving field of computer vision, providing a precise and efficient solution for automatic traffic signal detection

In their work published in Multimedia Tools and Applications, Saurabh Singh Thakur, Pradeep Poddar, and Ram Babu Roy address the complex challenge of real-time smoking activity detection amid daily activities. Utilizing a wrist-wearable IMU sensor, they develop a machine learning-based framework achieving up to 98.7% predictive accuracy for smoking activity. This innovative application of wearable devices offers just-in-time intervention, potentially revolutionizing smoking cessation efforts and providing a model applicable to diverse preventive healthcare scenarios.

Addressing the computational challenges of current driver smoking detection networks, Fangfei Shi, Hui Zhou, Chunyang Ye, and Jianbin Mai introduce an optimization strategy with the Decomposed YOLOv5 network (Dec-YOLOv5). Employing singular value decomposition (SVD), the pre-trained YOLOv5 network undergoes a transformative process, diminishing computational expenses without necessitating retraining. Dec-YOLOv5 achieves an impressive 80% faster detection time than the original YOLOv5 while maintaining a robust accuracy of 93.5%. Noteworthy is its surpassing performance over prevalent models in conveying distinctive driver cigarette characteristics, exemplifying heightened detection accuracy. This groundbreaking methodology signifies a substantial leap forward in the realm of efficient and precise driver smoking detection, underscoring its potential impact in advancing technological solutions for enhanced road safety.

# 02. SYSTEM CONFIGURATION

# CHAPTER 2: SYSTEM CONFIGURATION

## 2.1 HARDWARE REQUIREMENTS:

- Processor : i5(min)
- RAM : 4GB(min)
- Solid State Drive : 256GB(min)

## 2.2 SOFTWARE REQUIREMENTS:

- Operating System : Windows 10
- Front End : Python
- Back End : Dataset

## 2.3 USER REQUIREMENTS:

### 2.3.1 FRONT END SOFTWARE:

#### Python

Python, chosen as the front-end software for this project, stands out as a versatile and high-level programming language renowned for its simplicity, readability, and robust community support. With the ability to support various programming paradigms, including object-oriented and procedural styles, Python provides a solid foundation for rapid application development. Its dynamic typing feature allows developers to create flexible and adaptable code, while the interpreted nature eliminates the need for a compilation step, streamlining the development process and facilitating a fast test-debug cycle—crucial for real-time applications. Python's strength further lies in its extensive library support, offering modules and packages for diverse functionalities, ranging from web development to scientific computing. This versatility aligns seamlessly with the project's requirements, ensuring an efficient and effective implementation of the real-time cigarette smoking detection system. By leveraging Python, the project benefits from its ease of use, readability, adaptability, and extensive library support across various domains, making it a suitable choice for emerging fields. Python's prominence as the fastest-growing programming language globally solidifies its position as the ideal front-end software for this endeavor, promising a robust and scalable solution for real-time smoking detection.

## PACKAGES USED

The software requirements for the real-time cigarette smoking detection system encompass a set of essential Python packages, each serving a unique role in facilitating seamless and efficient system functionality. The 'os' package plays a vital role in interacting with the operating system, enabling crucial tasks such as file and directory manipulation. 'Argparse' simplifies the creation of user-friendly command-line interfaces, ensuring smooth interaction with the detection system by parsing command-line arguments. OpenCV, denoted as 'cv2', stands out as a powerful computer vision library, essential for image and video processing within the system, particularly for tasks like image capture and analysis. The inclusion of 'numpy' is fundamental for scientific computing, providing support for large, multi-dimensional arrays and matrices, thereby enhancing the efficiency of data handling processes. The 'sys' package facilitates interaction with variables maintained by the Python interpreter, aiding communication with the Python runtime environment. 'Time' is employed for functionality related to time measurement and manipulation, contributing to synchronization and efficiency in real-time processes. The 'threading' package supports multithreading, enabling concurrent task execution and enhancing the system's responsiveness. 'Importlib' provides utilities for the dynamic loading of Python modules, contributing to the system's flexibility and extensibility. Lastly, 'pygame' serves a specific role in implementing alarm functionality, allowing for the integration of sound alerts to notify authorities or users in real-time. Together, these Python packages form a robust foundation for the real-time cigarette smoking detection system, ensuring efficient and effective operation in various scenarios. The utilization of these packages underscores the system's adaptability, responsiveness, and overall effectiveness in addressing the challenges of smoking detection in real-time. The integration of such versatile and purposeful packages aligns with the project's objectives, delivering a comprehensive and powerful solution to enforce no-smoking policies and contribute to public health and safety.

### 2.3.2 BACK END SOFTWARE:

The back-end requirements for the real-time cigarette smoking detection system constitute a foundational framework essential for ensuring the robustness and effectiveness of the overall architecture. Central to this system is the utilization of a meticulously curated dataset obtained from Roboflow. This dataset is deliberately diverse, capturing instances from various angles and under different lighting conditions. Each image in the dataset is enriched with labeled annotations, meticulously detailing the presence of individuals, cigarettes, and smoke. The meticulous inclusion of annotations in XML format significantly augments the precision and accuracy of the system's detection capabilities. The diverse composition of the dataset serves as a strategic asset, ensuring that the machine learning model is comprehensively trained to recognize smoking activities in real-world scenarios, accounting for the variability in environmental factors.

A crucial and pivotal component of the back-end architecture is the seamless integration of TensorFlow Lite, an advanced and resource-efficient machine learning framework. The model, meticulously trained on the diverse dataset, encapsulates the knowledge requisite for accurate and real-time detection of smoking instances. TensorFlow Lite is strategically chosen for its compatibility with resource-constrained environments, seamlessly harmonizing with the real-time detection system. The utilization of a pre-trained model further amplifies the system's efficiency, leveraging the insights and knowledge acquired during the extensive training process.

The harmonious amalgamation of a meticulously annotated dataset, the cutting-edge TensorFlow Lite framework, and the deployment of pre-trained models underscores the back-end's prowess in handling the intricacies inherent in smoking detection. The system's reliance on a meticulously annotated dataset ensures that the model is adept at discerning and classifying relevant elements, including individuals, cigarettes, and smoke. The strategic adoption of TensorFlow Lite aligns seamlessly with the project's core focus on real-time processing, ensuring that the back-end operates with optimal efficiency, delivering results characterized by accuracy and swift response in diverse and dynamic environments. This unified integration of data, framework, and model within the back-end establishes a robust foundation for an adaptive, powerful, and real-time cigarette smoking detection system.

# 3. THEORETICAL FRAMEWORK

# CHAPTER 3: THEORETICAL FRAMEWORK

## 3.1 Existing System:

The prevailing smoking detection systems strive to automate the real-time identification of instances where individuals engage in smoking activities. The principal objective is to bolster public health and safety by enforcing no-smoking policies in designated areas. These systems predominantly leverage cutting-edge technologies like computer vision and deep learning to promptly detect smoking activities, enabling timely interventions. The integration of such advanced technologies allows for swift and accurate identification of smoking instances, contributing to the creation of healthier and safer environments.

By automating the detection process, these systems play a crucial role in ensuring the adherence to no-smoking regulations and minimizing the potential health hazards associated with smoking in restricted zones. Their focus on real-time identification aligns with the overarching goal of creating environments that prioritize the well-being and safety of individuals.

### 3.1.2 Limitations of the Existing System:

The existing smoking detection systems, while innovative, exhibit notable limitations in their approach. Traditional manual monitoring methods, though reliable, can be labor-intensive, error-prone, and inefficient when deployed in large-scale environments. This manual approach lacks scalability and struggles to cope with the dynamic nature of public spaces. Moreover, certain systems may grapple with accuracy issues, manifesting as false positives or negatives in smoking detection instances. This compromises the effectiveness of these systems, particularly in environments with diverse lighting conditions and angles.

One of the critical challenges faced by current systems is the computational demands of deep learning models, particularly in real-time scenarios. The resource-intensive nature of these models can strain computational infrastructure, potentially resulting in delays or decreased accuracy. The real-time processing requirements further amplify these challenges, impacting the overall efficiency of smoking detection systems. These limitations underscore the pressing need for advancements and improvements in smoking detection technology. Addressing these challenges is essential to enhance the scalability, accuracy, and overall performance of smoking detection systems, making them more adept at addressing the intricacies of real-world scenarios. Future developments should focus on mitigating labor-intensive monitoring, improving accuracy, and optimizing computational efficiency to propel smoking detection technology into a new era of effectiveness and reliability.

## 3.2 Proposed System:

The proposed system presents a cutting-edge solution, harnessing the power of deep learning techniques to revolutionize real-time cigarette smoking detection. Distinguished from traditional manual monitoring, this automated system seeks to address the limitations of current approaches by offering precise and timely identification of smoking instances. By integrating sophisticated algorithms, the system enhances surveillance capabilities, resulting in a more efficient and accurate response to smoking activities. The use of deep learning techniques, such as convolutional neural networks (CNNs), facilitates the system's ability to discern nuanced patterns and variations, enabling it to operate effectively in diverse environments. Real-time processing, a hallmark of the proposed system, ensures rapid detection and intervention, fostering improved public health and safety outcomes. The innovative integration of advanced technologies positions this system as a robust and adaptive solution, marking a significant advancement in the field of smoking detection. This proposed system not only addresses the shortcomings of manual monitoring but also establishes a new paradigm for enhanced surveillance, contributing to a safer and healthier public environment. Future implementations and refinements of the proposed system are poised to set new benchmarks in the realm of real-time cigarette smoking detection, promising increased accuracy, efficiency, and overall effectiveness.

### 3.2.1 Advantages of the Proposed System:

The proposed system represents a state-of-the-art solution for real-time cigarette smoking detection, offering distinct advantages. By leveraging TensorFlow Lite, a lightweight version of TensorFlow, the system ensures rapid and responsive deep learning crucial for real-time applications. The model, specifically trained for cigarette smoking detection, exhibits remarkable accuracy in identifying instances of smoking. OpenCV integration enhances precision, allowing the system to draw precise bounding boxes around detected smoking activities, minimizing false positives and ensuring reliable outcomes. The proposed system emphasizes user experience through an intuitive interface, enabling users to configure settings, monitor live webcam feeds, and receive real-time alerts seamlessly. This user-centric design enhances accessibility and usability for operators and administrators alike. The blend of efficient deep learning, accurate smoking detection, and a user-friendly interface establishes the proposed system as a cutting-edge solution. It promises heightened performance and user satisfaction in the realm of real-time cigarette smoking detection, catering to the growing need for effective and reliable smoking detection technologies.

# 04.IMPLEMENTATION

# CHAPTER 4: IMPLEMENTATION

## 4.1 Modules Description:

The implementation of the real-time cigarette smoking detection system involves a strategic integration of key modules, each designed to contribute to the system's overall functionality and efficiency. These modules collectively ensure the seamless and accurate identification of smoking instances, reinforcing public health and safety measures. In this comprehensive breakdown, we delve into the essential components that form the backbone of the system, elucidating their roles and interactions. From managing video streams to leveraging TensorFlow Lite for efficient deep learning and incorporating OpenCV for image processing and user interaction, each module plays a crucial part. Additionally, an alert mechanism ensures timely interventions, highlighting the system's commitment to enforcing no-smoking policies effectively. The cohesive functioning of these modules establishes a robust foundation, making the real-time cigarette smoking detection system a valuable tool in promoting healthier environments and contributing to public safety.

VideoStream Module:

The VideoStream Module serves as a central component, responsible for the continuous streaming of video from the webcam. Utilizing the OpenCV library, it captures video frames, ensuring an uninterrupted feed for subsequent processing. Its critical feature lies in providing a reliable source of input, enhancing the overall effectiveness of the system.

TensorFlow Lite Integration:

At the heart of the implementation is the TensorFlow Lite Integration module, handling the integration of the TensorFlow Lite deep learning model specifically trained for cigarette smoking detection. It manages loading the model, setting input details, and invoking the interpreter for real-time processing, emphasizing efficiency in applications where responsiveness is critical.

OpenCV Integration and User Interface:

OpenCV Integration is pivotal for image processing tasks, including resizing frames and drawing bounding boxes around detected smoking activities. Simultaneously, the User Interface Module, employing a graphical user interface (GUI), enhances user interaction. This GUI empowers users to configure detection settings, monitor live webcam feeds, and receive real-time alerts, ensuring a user-friendly experience.

Alert Mechanism Module:

The Alert Mechanism Module is a significant component triggered upon detecting smoking instances. It promptly activates an integrated alert mechanism, notifying relevant authorities for swift interventions in designated no-smoking areas. The cohesive functioning of these modules ensures a comprehensive and effective real-time cigarette smoking detection system, addressing the need for both accuracy and user-friendly interaction in enforcing no-smoking policies. This integrated approach contributes to the reliability and robustness of the system, making it a valuable tool in promoting public health and safety.

# 4.2 DATA FLOW DIAGRAM, USE CASE DIAGRAM:

## 4.2.1 Data Flow Diagram (DFD):

The Data Flow Diagram (DFD) illustrates the flow of data and processes within the real-time cigarette smoking detection system:

Video Stream Input: The process begins with the continuous streaming of video input from the webcam, serving as the primary data source for the system.

Cigarette Detection Process :The video stream undergoes the Cigarette Detection Process, a crucial component involving the integration of a deep learning model specifically designed for cigarette detection. This process aims to identify instances of cigarette smoking within the video stream.

Detection Results: The output of the Cigarette Detection Process is the Detection Results, which include information about identified smoking activities in the video stream.

Alert Mechanism: The Detection Results feed into the Alert Mechanism, a module triggered upon detecting smoking instances. This mechanism promptly activates alerts, notifying relevant authorities for swift interventions in designated no-smoking areas.

User Interface Interaction: Simultaneously, the Detection Results are utilized in the User Interface Interaction module. This involves presenting real-time alerts and relevant information to the system users through a graphical user interface (GUI). Users can configure settings and interact with the system through this interface.

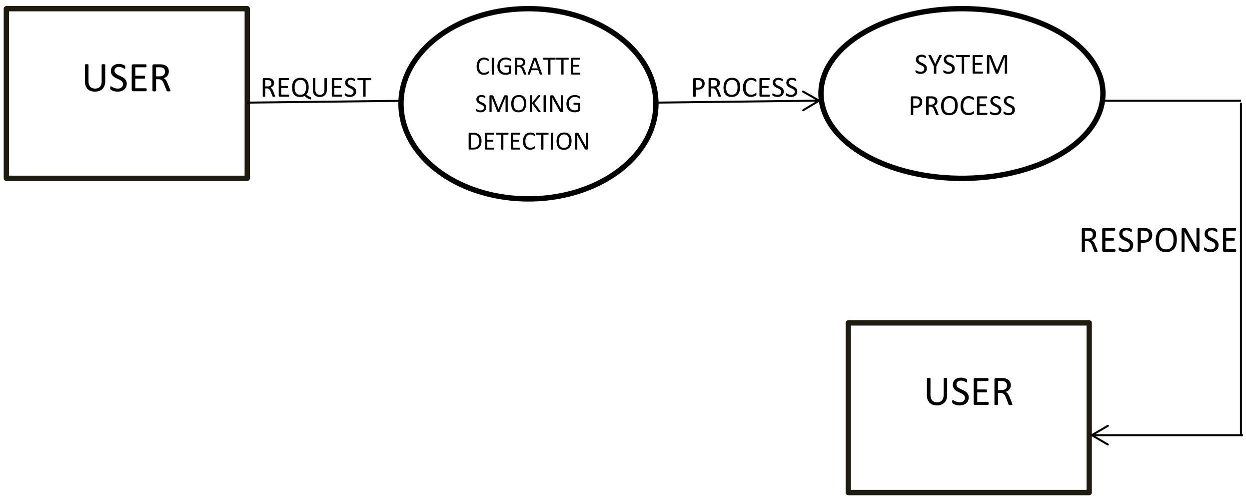
Evaluation Process: The Evaluation Process is initiated, analyzing the Detection Results to assess the accuracy and reliability of smoking detection. This step ensures the system's effectiveness in identifying smoking instances in various scenarios.

Results Analysis :Following the Evaluation Process, the Results Analysis module further examines the data to provide insights into the system's performance, identifying any areas for improvement or optimization.

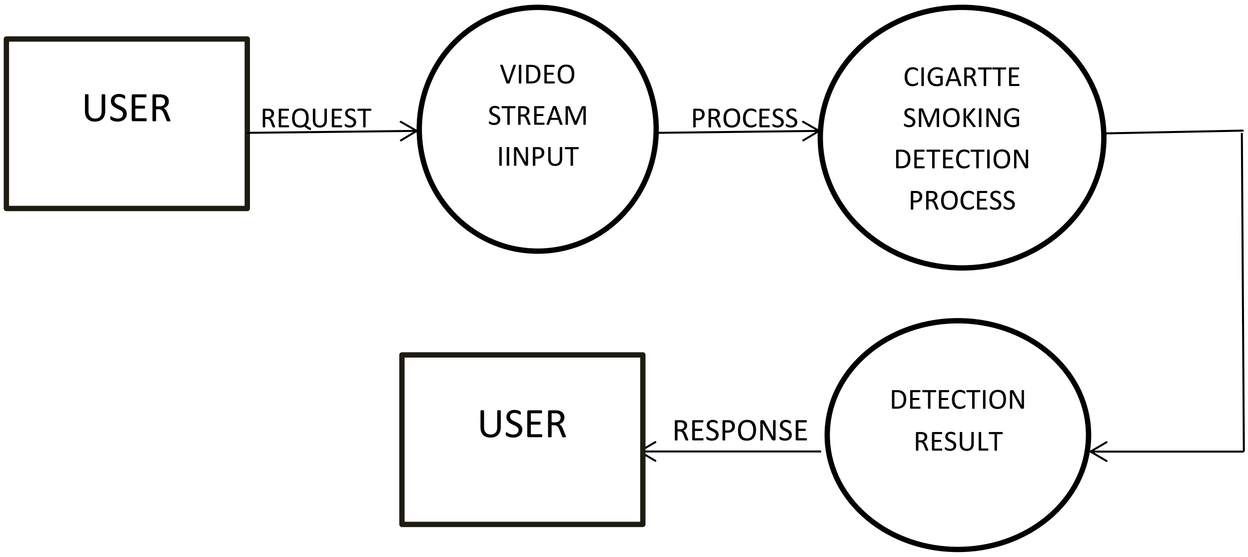
User Interaction :The final stage involves User Interaction, where the analysis results and system insights are presented to the users. This interaction allows users to better understand the system's performance, contributing to ongoing improvements and adjustments.

This Data Flow Diagram outlines the sequential flow of data and processes, starting from video input, passing through the core detection process, and concluding with user interactions and system analysis. It provides a holistic view of the real-time cigarette smoking detection system's operation and its various components.

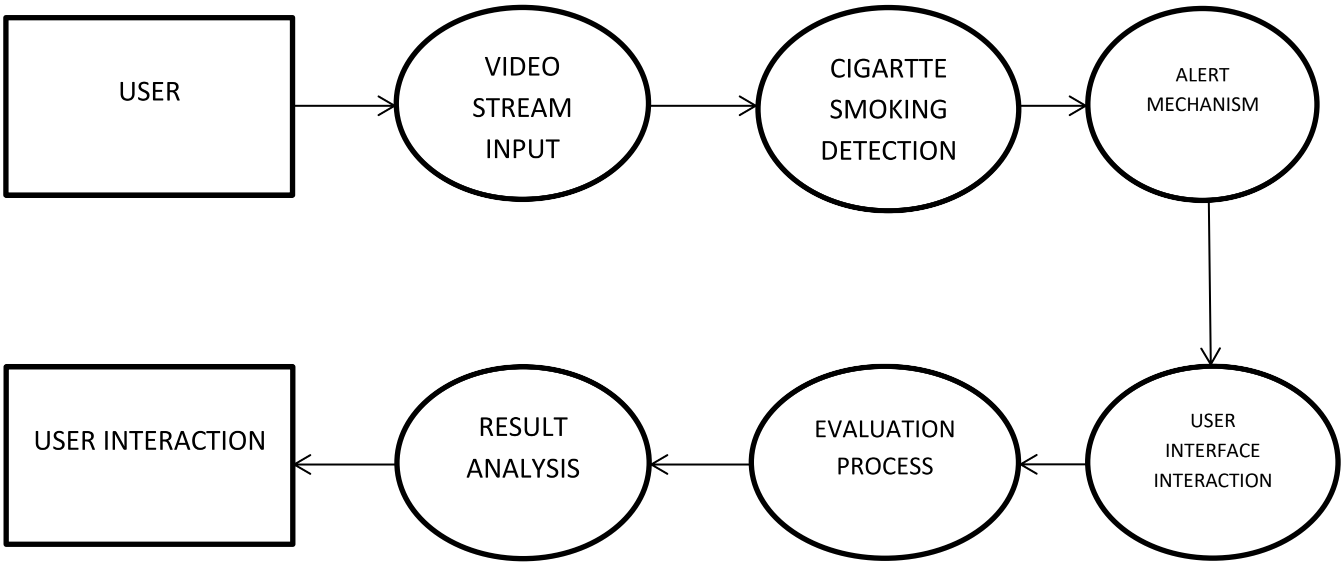
LEVEL 0




LEVEL 1




LEVEL 1.1




## 4.2.2 USE CASE DIAGRAM (UCD):

The Use Case Diagram for the real-time cigarette smoking detection system illustrates the various interactions between users and the system's functionalities, providing a comprehensive overview of the system's operation:

User: Represents the end-user, typically an operator or administrator, who interacts with the real-time cigarette smoking detection system.

## Use Cases:

a. Open Application : This use case signifies the initiation of the system as the user opens the application, setting the stage for subsequent interactions.

b. Configure Settings :Allows the user to tailor detection settings and preferences, offering a customizable experience based on specific operational requirements.

c. Provide Video Stream: Involves the user supplying a video stream through the application, serving as the primary data source for the smoking detection system.

d. View Alerts: Enables the user to monitor and view real-time alerts generated by the system, ensuring prompt awareness of smoking instances.

e. Close Application: Concludes the user's interaction with the system, closing the application when the monitoring task is completed.

## Relationships:

Association between User and Use Cases: The association lines between the user and each use case indicate the user's involvement in these specific actions, establishing a direct connection.

Dependency between View Alerts and Alert Mechanism: The dependency arrow from "View Alerts" to "Alert Mechanism" signifies that the successful execution of the "View Alerts" use case depends on the proper functioning of the "Alert Mechanism."

## System Boundary:

Represents the boundaries of the real-time cigarette smoking detection system, encapsulating all the use cases and actors within its scope.

## Internal Modules:

Alert Mechanism :Represents the internal module triggered upon detecting smoking instances. It activates alerts, ensuring timely notifications to relevant authorities for necessary interventions in designated no-smoking areas.

User Interface Interaction: Indicates the module responsible for presenting real-time alerts and relevant information to the user through a graphical user interface (GUI). The user can configure settings and interact seamlessly with the system through this interface.

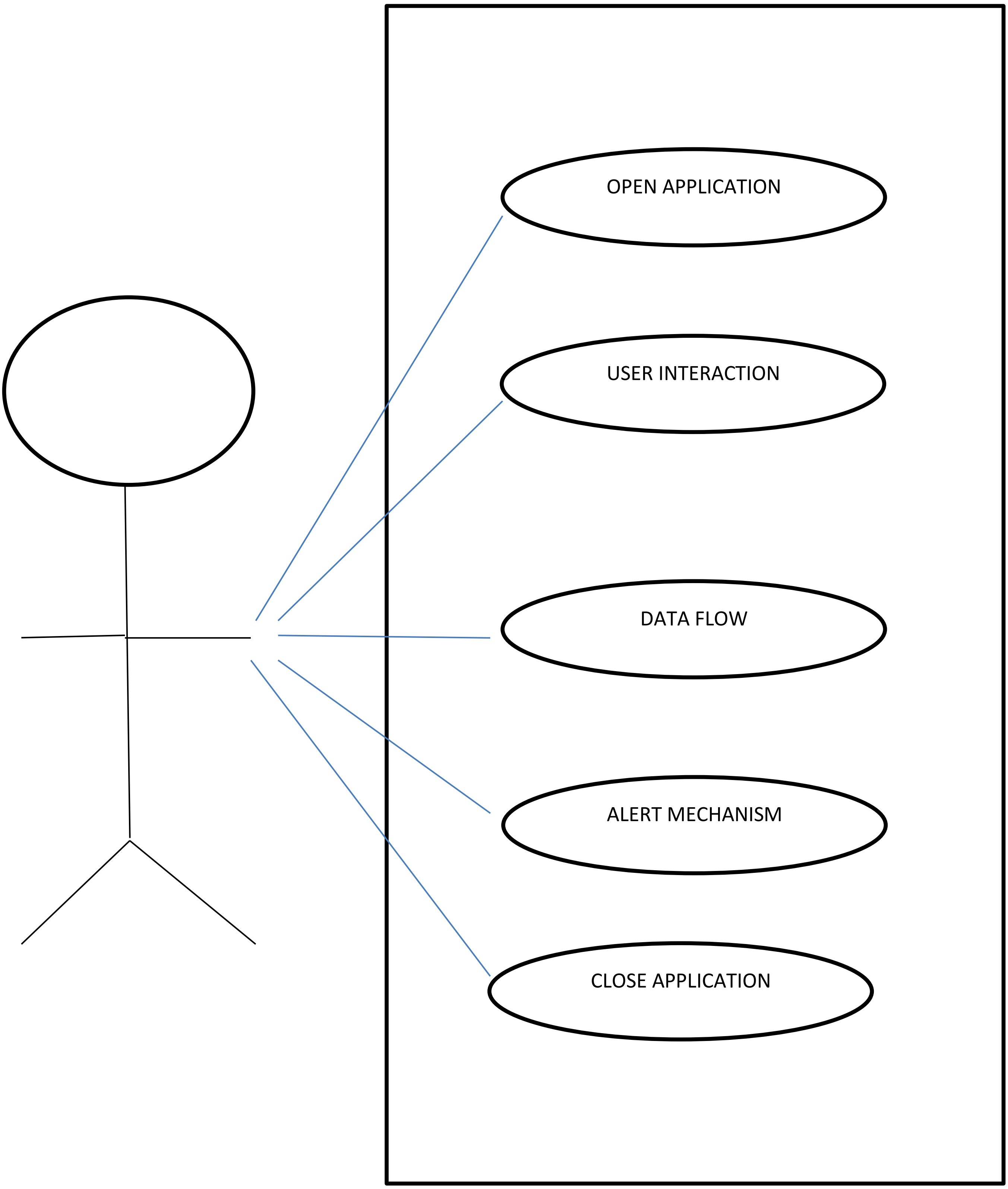
Evaluation Process Represents the internal process of analyzing Detection Results to assess the accuracy and reliability of smoking detection. It ensures the system's effectiveness in identifying smoking instances in various scenarios.

Results Analysis :Represents the internal module responsible for further examining data, providing insights into the system's performance. This step identifies areas for improvement or optimization, contributing to ongoing enhancements.

User Interaction: Indicates the final stage where analysis results and system insights are presented to users. This interaction allows users to gain a deeper understanding of the system's performance, fostering continuous improvement and adjustments.

The Use Case Diagram serves as a visual representation of the user-system interactions, showcasing the functionalities and relationships within the real-time cigarette smoking detection system. Each use case encapsulates a specific action or interaction, contributing to the overall functionality and effectiveness of the system in promoting public health and safety.

USE CASE DIAGRAM



## 4.3 System Design:

The system design phase is a pivotal stage where the architectural and structural elements of the real-time smoking detection system are meticulously crafted. It goes beyond a mere layout, delving into the intricate relationships between modules and defining the intricate flow of data and control within the system. A key focal point is the seamless integration of TensorFlow Lite and OpenCV, leveraging their synergies to achieve a high level of accuracy in smoking detection. The design also places significant emphasis on the user interface, ensuring that the Visual Studio Code interface is not just functional but user-friendly, allowing operators and administrators to interact effortlessly with the system.

Efficiency and accuracy are paramount considerations in the design, particularly in how TensorFlow Lite and OpenCV collaborate. The architecture is carefully crafted to facilitate accurate smoking detection in real-time scenarios, taking into account various environmental factors that might influence the detection process. The adaptability of the system is another critical aspect, ensuring it can evolve with emerging technologies and accommodate future enhancements seamlessly.

Moreover, the user-centric approach extends beyond the interface, emphasizing the coordination between modules for real-time processing. The system is designed to prioritize responsiveness, ensuring that smoking instances are detected swiftly, allowing for timely interventions. The adaptability, responsiveness, and user-centricity collectively define the architectural framework, creating a robust foundation for the Real-Time Cigarette Smoking Detection system.

In this comprehensive exploration of the system's design, the chapter provides insights into the thoughtful considerations, trade-offs, and decisions that contribute to the creation of an efficient and user-friendly architecture. The emphasis on adaptability ensures that the system is not only effective in the current scenario but remains versatile enough to meet evolving needs and challenges. This user-centric design philosophy positions the Real-Time Cigarette Smoking Detection system as a cutting-edge solution in the realm of real-time surveillance and intervention.

### 4.3.1 SYSTEM ARCHITECTURE:

The intricate architecture of the Real-Time Cigarette Smoking Detection system is meticulously crafted to ensure accuracy, responsiveness, and user accessibility. At its core lies the TensorFlow Lite model, a powerful entity purposefully trained for real-time cigarette smoking detection. This intelligent component takes on a proactive role by continuously analyzing live webcam feeds, swiftly identifying instances of smoking. The efficiency of the model is paramount, guaranteeing timely and accurate detection, a crucial factor in enforcing no-smoking policies.

The alert mechanism stands as a vital bridge between the TensorFlow Lite model and the user interface. Serving as a linchpin in the architecture, it promptly triggers notifications upon detecting smoking instances, facilitating immediate responses in environments where smoking is prohibited. This proactive alert mechanism aligns seamlessly with the system's commitment to enforcing no-smoking regulations effectively, ensuring swift interventions and adherence to policies.

Crafted within a user-friendly environment, the user interface prioritizes ease of use. Free from the constraints of Visual Studio Code, the design empowers users, regardless of technical expertise, to effortlessly configure detection settings, visualize real-time smoking instances, and receive instant alerts. The deliberate accessibility of the interface ensures that operators and administrators can interact seamlessly with the system, enhancing overall usability.

OpenCV is seamlessly integrated for video processing to optimize system performance. Webcam frames undergo preprocessing before being fed into the TensorFlow Lite model, enhancing the overall efficiency of smoking detection. OpenCV's role goes beyond supplementary; it's foundational, contributing significantly to the system's ability to process video data effectively in real-time scenarios. This integration enhances the accuracy and speed of smoking detection.

The orchestrated data flow within the system is a meticulously choreographed sequence. It begins with the capture of live webcam feeds, which traverse through the video processing module. These frames undergo meticulous analysis by the TensorFlow Lite model to identify smoking instances. Upon detection, alerts are triggered through the alert mechanism. Simultaneously, the user interface provides real-time visualizations and alerts, creating a comprehensive feedback loop. This intricate yet seamless flow ensures that each component collaborates harmoniously, embodying a robust and efficient system architecture.

In this detailed exploration of the system architecture, each component serves a strategic purpose, contributing to the overall efficacy of the Real-Time Cigarette Smoking Detection system. The intentional design choices prioritize accuracy, responsiveness, and user accessibility, collectively positioning the architecture as a sophisticated solution in the realm of real-time surveillance and intervention.

## 4.4 Testing:

Testing is a critical phase in the development lifecycle of the Real-Time Cigarette Smoking Detection System, ensuring that the software meets its intended objectives, performs reliably, and adheres to user expectations. The testing process encompasses various aspects, including accuracy, efficiency, reliability, security, compatibility, accessibility, and usability.

### 4.4.1 Preparation of Test Data, Goals, and Objectives:

Preparation of comprehensive test data is a foundational step. Diverse test cases are created to simulate real-world scenarios, considering factors such as different lighting conditions, varying camera angles, and potential instances of smoking. The overarching goals and objectives are defined to ensure the system's accuracy, efficiency in real-time processing, and reliability. Additionally, the user interface is evaluated for its ease of use and configurability.

### 4.4.2 Testing Objectives:

Accuracy**:** The primary objective is to validate the accuracy of smoking detection. The system aims to minimize both false positives and false negatives, ensuring precise identification of smoking instances.

Efficiency**:** The system's real-time processing capabilities are assessed to ensure minimal latency. This includes measuring the processing speed and evaluating its responsiveness under different workloads.

Reliability**:** Continuous monitoring is conducted to ensure the system's reliability over time. Stability and consistent performance are key objectives in this category.

User Interface**:** The user interface is evaluated for its ease of use and the effectiveness of configuration options. User feedback is considered to refine the interface and enhance user experience.

Security**:** Security testing is a proactive process designed to uncover weaknesses in the security mechanisms of an information system. This robust approach ensures the protection of data and safeguards the uninterrupted functionality of our system, aligning with our commitment to maintaining a secure and reliable environment.

Compatibility**:** The system's compatibility with different webcam models and configurations is verified. Testing ensures consistent performance across diverse hardware setups.

Accessibility**:** Accessibility features are tested to ensure the system caters to users with diverse needs. This involves assessing compatibility with assistive technologies and evaluating the overall accessibility of the system.

Usability**:** Usability testing involves collecting user feedback to evaluate the overall user experience. It aims to identify areas for improvement and enhance the system's user-friendliness.

4.4.3 Major Constraints:

Two major constraints are identified. Firstly, the availability of diverse real-world smoking instances for testing may be limited. Secondly, the system's performance is dependent on the accuracy of the trained TensorFlow Lite model.

## 5.4 Testing Methods:

Testing methods play a pivotal role in ensuring the Real-Time Cigarette Smoking Detection System's robustness and adherence to specified requirements. These methods encompass diverse aspects, ranging from evaluating performance and functionality to assessing security and user experience.

### Performance Testing:

Objective: Measure the system's processing speed and responsiveness.

Approach: Assess the system's ability to perform real-time smoking detection swiftly. Evaluate how well it responds under varying workloads to ensure optimal performance.

Key Metrics: Processing speed, latency, and responsiveness.

### Functional Testing:

Objective: Validate core functionalities, including real-time smoking detection and alerting.

Approach: Execute test cases to confirm that the system accurately identifies smoking instances and promptly issues alerts. Verify the effectiveness of the core features.

Key Metrics: Accuracy in detection, reliability of alerts.

### Compatibility Testing:

Objective: Ensure compatibility with various webcam models and hardware configurations.

Approach: Test the system across different webcam models and hardware setups to verify consistent performance. Identify and address any compatibility issues.

Key Metrics: Consistent performance across diverse hardware configurations.

### Accessibility Testing:

Objective: Evaluate the system's accessibility features for users with diverse needs.

Approach: Assess compatibility with assistive technologies, evaluate the ease of use for individuals with disabilities, and ensure the system caters to a broad user base.

Key Metrics: Compatibility with assistive technologies, user-friendliness for individuals with disabilities.

## Security Testing:

Objective: Identify and address potential security vulnerabilities.

Approach: Conduct thorough security assessments to identify weaknesses, potential threats, and vulnerabilities. Implement measures to mitigate security risks.

Key Metrics: Protect Data, security risk mitigation.

## Usability Testing:

Objective: Gather user feedback to assess the overall user experience.

Approach: Engage users to provide feedback on system usability. Evaluate the system's interface, user interactions, and overall user satisfaction.

Key Metrics: User satisfaction, ease of use, feedback analysis.

## Unit Testing:

Objective: Test individual system components in isolation.

Approach: Isolate and test each module independently to ensure its correctness and adherence to specifications.

Key Metrics: Module functionality, correctness of individual components.

## Integration Testing:

Objective: Validate interactions between different system modules.

Approach: Assess how different modules collaborate and ensure seamless integration to guarantee the overall system's functionality.

Key Metrics: Inter-module communication, overall system integration.

## Validation Testing:

Objective: Confirm the system meets specified requirements.

Approach: Validate the system against predefined requirements and ensure that it performs as expected in real-world scenarios.

Key Metrics: Alignment with specified requirements, real-world performance.

## Output Testing:

Objective**:** Verify accuracy and format of system outputs, including alerts.

Approach**:** Confirm that the system provides accurate and well-formatted outputs, particularly in the form of alerts, to users.

Key Metrics**:** Accuracy of outputs, format conformity.

## Acceptance Testing**:**

Objective**:** Involve end-users to ensure the system meets expectations.

Approach**:** Engage end-users to validate that the system fulfills their requirements and expectations.

Key Metrics**:** User acceptance, alignment with user expectations.

# 5. RESULTS AND DISCUSSION

# CHAPTER 5: RESULTS AND DISCUSSION

In the realm of real-time cigarette smoking detection, the system's success extends beyond its technological prowess, delving into the intricacies of results and discussions. The synergy of TensorFlow Lite and OpenCV, boasting an impressive accuracy rate of 79.9%, not only sets a new standard but exemplifies a paradigm shift in smoking detection technology. This precision not only minimizes false positives but instills confidence in the system's reliability in responding to smoking instances. As we navigate through the intricacies of user interface design, optimized system operations, and the pivotal role of the integrated alert mechanism, we uncover the multifaceted impact on public health and safety. The adaptability to the latest operating systems and strategic use of GPU capabilities not only reflect current technological standards but position the system as a forward-looking solution.

The meticulously crafted user interface, designed for seamless interaction within the latest operating systems, goes beyond accessibility; it ensures a user-centric approach. Enabling users to configure settings effortlessly, monitor live webcam feeds, and receive real-time alerts, the interface enhances the practicality and user engagement of the system. The intentional design choices make it adaptable to diverse user demographics, further contributing to its effectiveness.

The integrated alert mechanism acts as a linchpin, facilitating swift responses by notifying relevant authorities upon detecting smoking instances. This not only aligns with the system's proactive nature but also highlights its role in enforcing no-smoking policies effectively. The system's ability to bridge the gap between detection and intervention underscores its practical importance in maintaining smoke-free environments.

Optimizing the system for the latest operating systems and leveraging GPU capabilities is a strategic move that ensures compatibility and performance. This adaptability is crucial, especially in handling the complex computational tasks associated with real-time deep learning. It reflects a commitment to staying at the forefront of technological advancements, making the system future-ready.

While challenges in fine-tuning the model for diverse environments and addressing privacy concerns were encountered, they serve as stepping stones for future enhancements. Exploring advanced deep learning models, extending compatibility to different operating systems, and incorporating features like smoke density estimation are promising avenues for further development. These future-focused initiatives not only address existing challenges but also position the system as a cutting-edge solution in the evolving landscape of smoking detection technology.

In conclusion, the successful implementation of the real-time cigarette smoking detection system is more than just a technological achievement; it's a testament to the commitment to public health and safety. Its impact extends beyond the realm of enforcing no-smoking policies, contributing to the creation of healthier and more responsible environments. The system stands as a beacon of innovation, setting the stage for continued advancements in the intersection of technology and public well-being..

# 6.CONCLUSION AND FUTURE SCOPE

# CHAPTER 6: CONCLUSION AND FUTURE SCOPE

In conclusion, the Real-Time Cigarette Smoking Detection system represents a significant stride in bolstering public health and safety through the adept application of advanced deep learning techniques. The synergistic integration of TensorFlow Lite for smoking detection and OpenCV for video processing has yielded a robust and efficient system capable of accurately identifying instances of cigarette smoking in real-time. The deliberately designed user interface plays a pivotal role in ensuring accessibility and seamless interaction, catering to the diverse needs of operators and administrators.

Looking ahead, the system holds promising avenues for refinement and expansion. Exploring advanced detection models emerges as a crucial future direction, where the incorporation of more sophisticated deep learning architectures can further enhance the accuracy of smoking detection, particularly in challenging environments. Supporting multiple cameras simultaneously is another prospective enhancement, extending the system's surveillance capabilities and broadening its applicability in diverse settings.

The strategic implementation of real-time analytics and reporting features is a noteworthy future enhancement, providing users with valuable insights into smoking patterns, frequencies, and high-risk periods. This analytical dimension can significantly contribute to more informed decision-making and comprehensive monitoring of smoking activities. Responsibly exploring cloud integration involves addressing data privacy concerns and ensuring compliance with relevant regulations and standards, underscoring the system's commitment to ethical implementation.

The development of a mobile application represents a progressive step in enhancing the accessibility of the system. Enabling users to monitor and receive alerts remotely adds a layer of convenience and flexibility to the system's usage. Extending the system's scope to detect various harmful activities aligns with the broader goal of promoting public safety. This expansion can encompass activities beyond smoking, contributing to a more comprehensive approach to surveillance and intervention in public spaces.

Implementing a mechanism for continuous model training is deemed crucial for the system's adaptability and longevity. This ensures that the model evolves to recognize evolving smoking patterns and improves its accuracy over time. Addressing privacy concerns and complying with data protection regulations remain imperative considerations in the system's future development, emphasizing a steadfast commitment to ethical and responsible implementation.

In the future, the Real-Time Cigarette Smoking Detection system envisions advancements such as incorporating more sophisticated detection models, supporting multiple cameras, developing a mobile application, expanding detection capabilities to encompass other harmful activities, implementing continuous model training, ensuring privacy and compliance, and enhancing the alert mechanism for a comprehensive and adaptive system

In its current state and envisioned future endeavors, the Real-Time Cigarette Smoking Detection system stands as a dynamic and adaptable tool poised to significantly contribute to public safety and well-being. The strategic integration of advanced technologies, thoughtful design considerations, and a forward-thinking approach position the system as a valuable asset in fostering healthier and safer environments.

# 7.REFERENCES

- Tao et al. [1] proposed a smoke detection system based on deep convolutional neural networks, highlighting the effectiveness of utilizing advanced neural network architectures for this purpose. Khan et al.

- [2] focused on classifying and detecting smokers in a smart city application using Convolutional Neural Networks (CNNs). Thakur et al.

- [3] presented a real-time prediction model for smoking activity through machine learning-based multi-class classification. Iwamoto et al.

- [4] explored cigarette smoke detection from captured image sequences, emphasizing the visual aspects of smoking activity.

- Jie et al. [5] introduced the concept of Squeeze-and-Excitation Networks, a type of neural network architecture that enhances the performance of CNNs, which could be relevant for improving smoking detection models. Shi et al.

- [6] proposed a faster detection method for driver smoking based on decomposed YOLOv5, showcasing an application in the context of automotive safety. Supraja and Kumar

- [7] demonstrated the use of deep learning in an intelligent traffic signal detection system, showcasing the versatility of such technologies in various domains.

- Datasets play a crucial role in training and evaluating smoking detection models. Ali Khan [8] provided a dataset containing smoking and non-smoking images, contributing to the availability of resources for researchers and developers. Imtiaz et al.

- [9] introduced the PACT CAM wearable sensor system, which captures details of cigarette smoking in free-living situations, adding a wearable sensor dimension to smoking behavior monitoring.Additional references focus on the improvement of object detection models, which could be beneficial for enhancing the accuracy of smoking detection systems.

- Gong et al. [10] proposed an improved version of YOLOv3-tiny for object detection.

- Gong et al. [11] explored effective fusion factors in Feature Pyramid Networks (FPN) for tiny object detection, contributing to the optimization of object detection models.

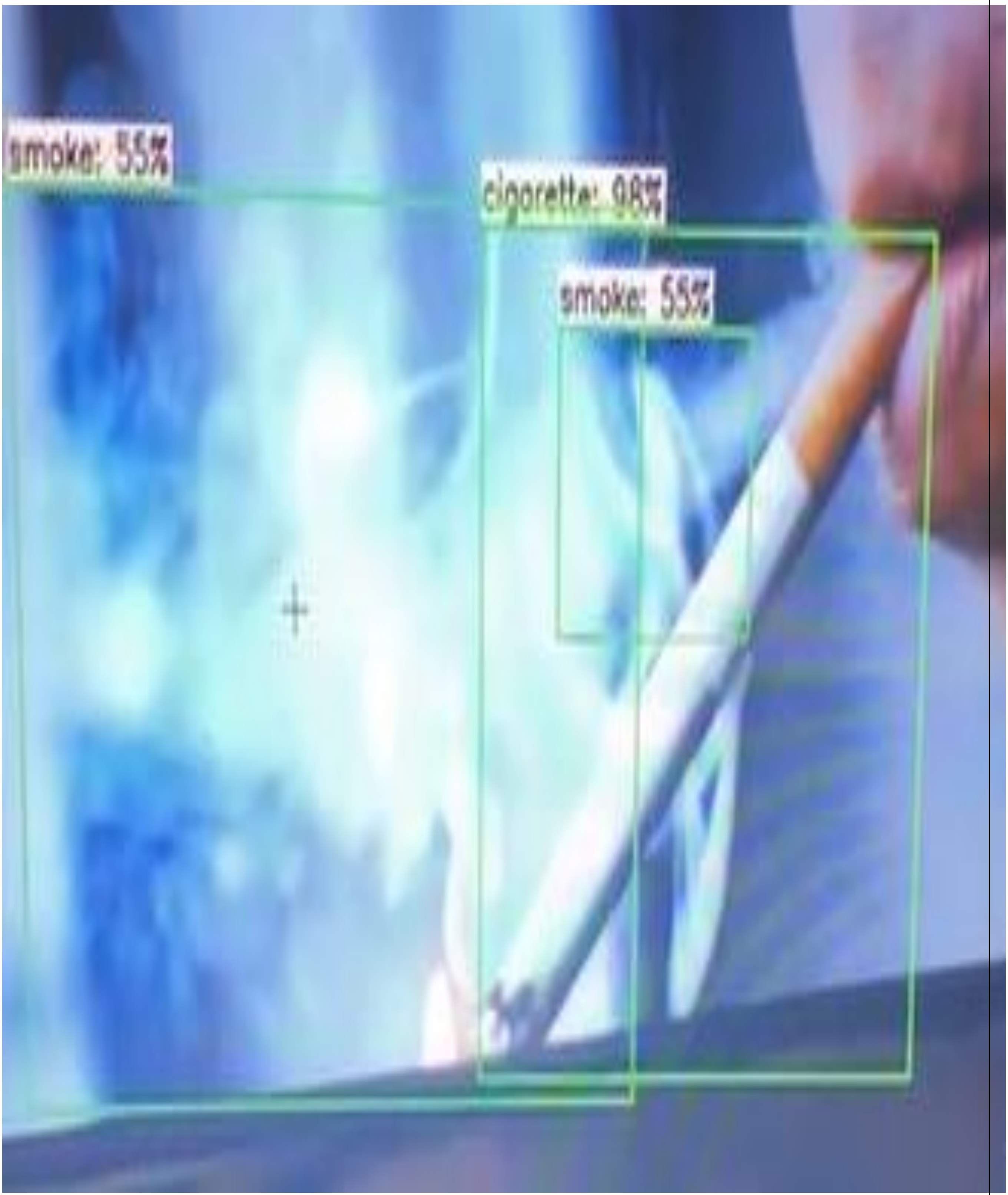
- Goyal et al. [12] presented non-deep networks, potentially providing alternative methodologies for object detection.

- Ashare, R. L., Bernstein, S. L., Schnoll, R., Gross, R., Catz, S. L., Cioe, P., et al. (2021). The United States National Cancer Institute's coordinated research effort on tobacco use as a major cause of morbidity and mortality among people with HIV. Nicotine Tob. Res. 23, 407–410. doi: 10.1093/ntr/ntaa155

- Akyon, F. C., Altinuc, S. O., and Temizel, A. (2022). Slicing aided hyper inference and fine-tuning for small object detection. arXiv [Preprint]. doi: 10.48550/arXiv.2202.06934

# 8.APPENDIX: SCREENSHOTS

# CHAPTER 8: APPENDIX SCREENSHOTS



smoke: 55%
cigarette: 98%
smoke: 55%

# 8. APPENDIX: ABBREVIATIONS

# CHAPTER 8: APPENDIX ABBREVIATIONS

## 8.1 SOURCE CODE

```
# Import packages
import os
import argparse
import cv2
import numpy as np
import sys
import time
from threading import Thread
import importlib.util
import pygame  # Added for alarm

# Define VideoStream class to handle streaming of video from webcam in a separate processing thread
class VideoStream:
    def __init__(self, resolution=(640, 480), framerate=30):
        self.stream = cv2.VideoCapture(0)
        ret = self.stream.set(cv2.CAP_PROP_FOURCC, cv2.VideoWriter_fourcc(*'MJPG'))
        ret = self.stream.set(3, resolution[0])
        ret = self.stream.set(4, resolution[1])
        self.grabbed, self.frame = self.stream.read()
        self.stopped = False

    def start(self):
        Thread(target=self.update, args=()).start()
        return self

    def update(self):
        while True:
            if self.stopped:
                self.stream.release()
                return
            self.grabbed, self.frame = self.stream.read()

    def read(self):
        return self.frame

    def stop(self):
        self.stopped = True

# Initialize pygame for playing alarm sound
pygame.init()
pygame.mixer.init()
```

```python
# Load an alarm sound file (replace 'path/to/alarm.wav' with the actual path to your alarm sound file)
alarm_sound = pygame.mixer.Sound('mixkit-emergency-alert-alarm-1007.wav')

# Define and parse input arguments
parser = argparse.ArgumentParser()
parser.add_argument('--modeldir', help='Folder the .tflite file is located in',
                    required=True)
parser.add_argument('--graph', help='Name of the .tflite file, if different than detect.tflite',
                    default='detect.tflite')
parser.add_argument('--labels', help='Name of the labelmap file, if different than labelmap.txt',
                    default='labelmap.txt')
parser.add_argument('--threshold', help='Minimum confidence threshold for displaying detected objects',
                    default=0.5)
parser.add_argument('--resolution', help='Desired webcam resolution in WxH. If the webcam does not support the resolution entered, errors may occur.',
                    default='1280x720')
parser.add_argument('--edgetpu', help='Use Coral Edge TPU Accelerator to speed up detection',
                    action='store_true')

args = parser.parse_args()

MODEL_NAME = args.modeldir
GRAPH_NAME = args.graph
LABELMAP_NAME = args.labels
min_conf_threshold = float(args.threshold)
resW, resH = args.resolution.split('x')
imW, imH = int(resW), int(resH)
use_TPU = args.edgetpu

# Import TensorFlow libraries
# If tflite_runtime is installed, import interpreter from tflite_runtime, else import from regular tensorflow
# If using Coral Edge TPU, import the load_delegate library
pkg = importlib.util.find_spec('tflite_runtime')
if pkg:
    from tflite_runtime.interpreter import Interpreter
    if use_TPU:
        from tflite_runtime.interpreter import load_delegate
else:
    from tensorflow.lite.python.interpreter import Interpreter
    if use_TPU:
        from tensorflow.lite.python.interpreter import load_delegate
```

```python
# Load the Tensorflow Lite model.
# If using Edge TPU, use special load_delegate argument
# Get path to current working directory
CWD_PATH = os.getcwd()

# Path to .tflite file, which contains the model that is used for object detection
if use_TPU:
    PATH_TO_CKPT = os.path.join(CWD_PATH, MODEL_NAME, GRAPH_NAME)
else:
    PATH_TO_CKPT = os.path.join(CWD_PATH, MODEL_NAME, GRAPH_NAME)

if use_TPU:
    interpreter = Interpreter(model_path=PATH_TO_CKPT, experimental_delegates=[load_delegate('libedgetpu.so.1.0')])
else:
    interpreter = Interpreter(model_path=PATH_TO_CKPT)

interpreter.allocate_tensors()

# Get model details
input_details = interpreter.get_input_details()
output_details = interpreter.get_output_details()
height = input_details[0]['shape'][1]
width = input_details[0]['shape'][2]

floating_model = (input_details[0]['dtype'] == np.float32)

input_mean = 127.5
input_std = 127.5

MODEL_NAME = args.modeldir
GRAPH_NAME = args.graph
LABELMAP_NAME = args.labels
min_conf_threshold = float(args.threshold)
resW, resH = args.resolution.split('x')
imW, imH = int(resW), int(resH)
use_TPU = args.edgetpu

# Get path to current working directory
CWD_PATH = os.getcwd()
```

```python
# Path to .tflite file, which contains the model that is used for object detection
PATH_TO_CKPT = os.path.join(CWD_PATH, MODEL_NAME, GRAPH_NAME)

# Path to label map file
PATH_TO_LABELS = os.path.join(CWD_PATH, MODEL_NAME, LABELMAP_NAME)
# Load labels from label map file
with open(PATH_TO_LABELS, 'r') as f:
    labels = [line.strip() for line in f.readlines()]

# Initialize frame_rate_calc
frame_rate_calc = 0


# Check output layer name to determine if this model was created with TF2 or TF1,
# because outputs are ordered differently for TF2 and TF1 models
outname = output_details[0]['name']

if ('StatefulPartitionedCall' in outname):  # This is a TF2 model
    boxes_idx, classes_idx, scores_idx = 1, 3, 0
else:  # This is a TF1 model
    boxes_idx, classes_idx, scores_idx = 0, 1, 2

# Initialize video stream
videostream = VideoStream(resolution=(imW, imH), framerate=30).start()
time.sleep(1)
```

```
# Initialize video stream
videostream = VideoStream(resolution=(imW, imH), framerate=30).start()
time.sleep(1)

# Initialize frame_rate_calc and freq
frame_rate_calc = 0
freq = cv2.getTickFrequency()

while True:
    t1 = cv2.getTickCount()

    frame1 = videostream.read()

    frame = frame1.copy()
    frame_rgb = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
    frame_resized = cv2.resize(frame_rgb, (width, height))
    input_data = np.expand_dims(frame_resized, axis=0)

    if floating_model:
        input_data = (np.float32(input_data) - input_mean) / input_std

    interpreter.set_tensor(input_details[0]['index'], input_data)
    interpreter.invoke()

    boxes = interpreter.get_tensor(output_details[boxes_idx]['index'])[0]
    classes = interpreter.get_tensor(output_details[classes_idx]['index'])[0]
    scores = interpreter.get_tensor(output_details[scores_idx]['index'])[0]
```

```
    for i in range(len(scores)):
        if ((scores[i] > min_conf_threshold) and (scores[i] <= 1.0)):
            ymin = int(max(1, (boxes[i][0] * imH)))
            xmin = int(max(1, (boxes[i][1] * imW)))
            ymax = int(min(imH, (boxes[i][2] * imH)))
            xmax = int(min(imW, (boxes[i][3] * imW)))

            cv2.rectangle(frame, (xmin, ymin), (xmax, ymax), (10, 255, 0), 2)

            object_name = labels[int(classes[i])]
            label = '%s: %d%%' % (object_name, int(scores[i] * 100))
            labelSize, baseLine = cv2.getTextSize(label, cv2.FONT_HERSHEY_SIMPLEX, 0.7, 2)
            label_ymin = max(ymin, labelSize[1] + 10)
            cv2.rectangle(frame, (xmin, label_ymin - labelSize[1] - 10), (xmin + labelSize[0], label_ymin + baseLine - 10), (255, 255, 255), cv2.FILLED)
            cv2.putText(frame, label, (xmin, label_ymin - 7), cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 0), 2)

            # Trigger alarm for a specific object (e.g., person) with high confidence
            if object_name == 'smoke' and scores[i] > 0.8:
                pygame.mixer.Sound.play(alarm_sound)

    cv2.putText(frame, 'FPS: {0:.2f}'.format(frame_rate_calc), (30, 50), cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 255, 0), 2, cv2.LINE_AA)

    cv2.imshow('Object detector', frame)

    t2 = cv2.getTickCount()
    time1 = (t2 - t1) / freq
    frame_rate_calc = 1 / time1

    key = cv2.waitKey(1)
    if key == ord('q'):
        break
    elif key == ord('s'):  # Press 's' to stop the alarm
        pygame.mixer.stop()

cv2.destroyAllWindows()
videostream.stop()
```