A PROJECT REPORT

on

"EFFICIENT INDOOR CROWD MANAGEMENT SYSTEM"

Submitted to

KIIT Deemed to be University

In Partial Fulfillment of the Requirement for the Award of

BACHELOR'S DEGREE IN
COMPUTER SCIENCE AND ENGINEERING

BY

SANSKAR JAIN	20051164
SWEEKRITI PANT	20051182
B. SARANDEEP SABUT	20051397
AASHISH KUMAR	20051433

UNDER THE GUIDANCE OF
DR. RAJDEEP CHATTERJEE
(Associate Professor)

SCHOOL OF COMPUTER ENGINEERING
KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY
BHUBANESWAR, ODISHA – 751024

May 2023

A PROJECT REPORT

on

"EFFICIENT INDOOR CROWD MANAGEMENT SYSTEM"

Submitted to

KIIT Deemed to be University

In Partial Fulfillment of the Requirement for the Award of

BACHELOR'S DEGREE IN

COMPUTER SCIENCE AND ENGINEERING

BY

SANSKAR JAIN	20051164
SWEEKRITI PANT	20051182
B. SARANDEEP SABUT	20051397
AASHISH KUMAR	20051433

UNDER THE GUIDANCE OF

DR. RAJDEEP CHATTERJEE

(Associate Professor)

SCHOOL OF COMPUTER ENGINEERING

KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY

BHUBANESWAE, ODISHA -751024

May 2023

KIIT Deemed to be University

School of Computer Engineering
Bhubaneswar, ODISHA 751024

CERTIFICATE

This is to certify that the project entitled

"EFFICIENT INDOOR CROWD MANAGEMENT SYSTEM"

submitted by

SANSKAR JAIN	20051164
SWEEKRITI PANT	20051182
B. SARANDEEP SABUT	20051397
AASHISH KUMAR	20051433

is a record of bonafide work carried out by them, in the partial fulfillment of the requirement for the award of Degree of Bachelor of Engineering (Computer Science & Engineering) at KIIT Deemed to be university, Bhubaneswar. This work is done during year 2022-2023, under our guidance.

Date: 01/05/23

(Dr. Radjeep Chatterjee)
Project Guide

Acknowledgements

We are profoundly grateful to **DR. RAJDEEP CHATERJEE** of **School of Computer Engineering, KIIT Deemed to be University** for his expert guidance and continuous encouragement throughout to see that this project rights its target since its commencement to its completion. .....

SANSKAR JAIN
SWEEKRITI PANT
B. SARANDEEP SABUT
AASHISH KUMAR

ABSTRACT

Efficient Indoor Crowd Management is a critical aspect of ensuring Public Safety and optimizing Crowd Control in various environments such as shopping malls, event venues, and transportation hubs. This project aims to develop a sophisticated Crowd Management System using YOLO (You Only Look Once) Object Detection algorithm and Flask Web Framework. The system utilizes Computer Vision techniques to detect and track individuals in real-time, enabling accurate crowd counting and monitoring. By providing real-time insights into crowd density and movement patterns, the system facilitates Efficient Crowd Management and enhances overall security measures.

The proposed system leverages the power of YOLO, an advanced object detection model, to accurately identify and track individuals within a video stream. By applying this algorithm to live camera feeds, the system is able to detect and count the number of people present in a given staircase. The captured data is then processed and displayed in a User-friendly Web Interface created using Flask, allowing authorized personnel to monitor crowd levels and make informed decisions regarding crowd control measures. Additionally, the system incorporates features like color-coded threshold alerts, where the display changes to red when the crowd density exceeds a predefined threshold, providing visual cues for immediate action and redirecting the crowd to the other staircase. By continuously refining the algorithm and expanding the system's capabilities, it has the potential to be deployed in various domains, ranging from public safety and security to crowd analytics for business optimization. This combination of YOLO-based object detection and Flask web framework presents a powerful solution for Real-Time Crowd Monitoring and opens avenues for future research and advancements in the field.

Keywords: Crowd Management, Object Detection, Real-time monitoring, Indoor security, YOLO algorithm, Flask web framework, Crowd Counting, Public Safety, Computer Vision, Machine Learning.

Contents

List of Figures

Chapter 1

Introduction

Indoor crowd management has emerged as a critical concern in various public spaces, including shopping malls, airports, and hospitals. Ensuring the safety and comfort of visitors necessitates effective crowd management systems capable of monitoring crowd density and promptly alerting authorities in case of overcrowding. In this paper, we present a novel and efficient indoor crowd management system model developed using Python.

Our proposed model harnesses the power of image processing techniques and machine learning algorithms to address the challenges associated with crowd management. By deploying web cameras strategically in different areas of the public space, real-time footage of the crowd is captured. This footage undergoes meticulous processing using sophisticated computer vision algorithms that detect and track individuals within the frame, enabling accurate estimation of crowd density. Subsequently, a machine learning algorithm utilizes the current crowd density information to predict the crowd density for the upcoming minutes, empowering proactive measures to be taken.

The implementation of our proposed model entails three crucial steps: data collection, data processing, and data analysis. During the data collection phase, footage captured by the web cameras is provided frame-by-frame in real-time for subsequent processing. In the data processing phase, an advanced computer vision algorithm analyzes the footage to detect people and estimate crowd density. The processed data is then stored for further analysis. Lastly, the data analysis phase employs machine learning algorithms to analyze the stored data, enabling accurate

prediction of crowd density for the near future. In case of overcrowding, the model promptly alerts the authorities, facilitating swift intervention.

Our proposed indoor crowd management system model developed in Python showcases the fusion of two pivotal technologies: computer vision and machine learning.

Computer Vision: This technology is used to analyze the footage captured by web cameras and detect the number of people in the frame. The computer vision algorithms also estimate the crowd density in real-time. These algorithms use image processing techniques to detect and track objects, such as people, within the frame.

Machine Learning: This technology is used to predict the crowd density for the next few minutes based on the current crowd density. The model is trained using historical data to learn the patterns and trends in the crowd density. The machine learning algorithms used in this model are based on regression and time-series analysis techniques.

Our Indoor crowd management system leverages cutting-edge technologies and intelligent algorithms to provide real-time crowd monitoring, analysis, and proactive decision-making capabilities. By utilizing a network of strategically placed sensors and cameras, the system captures and analyzes data regarding crowd movement, density, and behavior. Advanced computer vision techniques are employed to accurately detect and track individuals, allowing for precise crowd density estimation. This information is then processed and analyzed using machine learning algorithms, which leverage historical data and patterns to predict crowd behavior and anticipate potential overcrowding situations.

One of the key advantages of our indoor crowd management system is its ability to provide actionable insights and real-time alerts to authorities and stakeholders. By continuously monitoring crowd density and movement patterns, the system can detect anomalies and trigger alerts when predefined thresholds are exceeded. These alerts enable timely interventions, such as crowd flow redirection, capacity adjustments, or the deployment of additional personnel to ensure crowd safety and optimal space utilization. Moreover, our system can generate detailed reports and visualizations that offer valuable insights into crowd behavior, helping authorities make informed decisions for future planning, resource allocation, and infrastructure improvements.

By introducing this advanced indoor crowd management system, we aim to revolutionize the way public spaces are managed and create safer, more comfortable environments for individuals. Through the integration of state-of-the-art technologies, our system empowers authorities to monitor and respond to crowd dynamics in real-time, mitigating potential risks and enhancing crowd management strategies. With its scalable architecture and adaptability, the system can be implemented in various settings, including shopping malls, stadiums, transportation hubs, and other crowded spaces. Our solution sets the stage for a new era of intelligent crowd management, where data-driven insights and proactive measures ensure the well-being and satisfaction of all individuals in public environments.

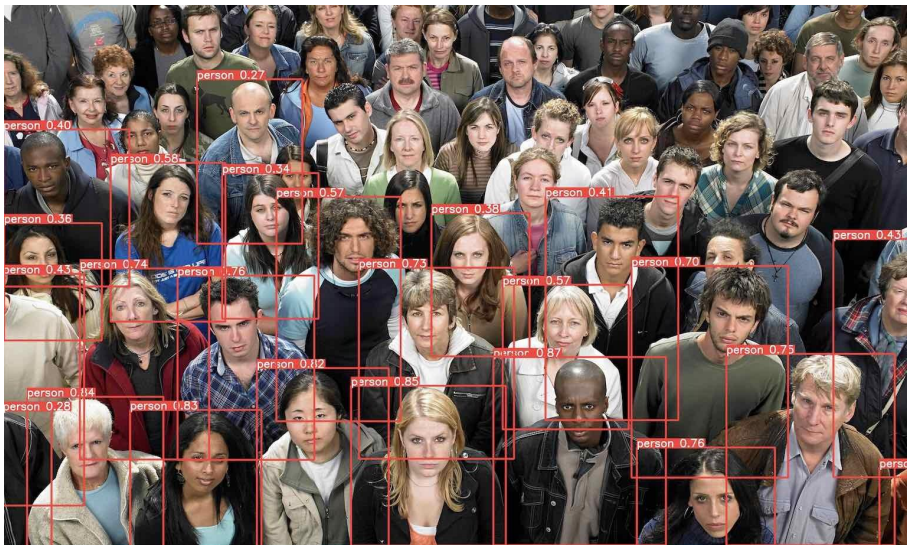


Fig 1.1 (Crowd Detection)

Chapter 2

Basic Concepts

2.1 HTML

HTML stands for Hyper Text Markup Language which is used for creating web pages and web applications.

Hypertext: Hypertext means "Text within Text." A text has a link within it, is a hypertext. Whenever you click on a link which brings you to a new webpage, you have clicked on a hypertext. Hypertext is a way to link two or more web pages (HTML documents) with each other.

Markup language: A markup language is a computer language that is used to apply layout and formatting conventions to a text document. Markup language makes text more interactive and dynamic. It can turn text into images, tables, links, etc.

Web Page: A web page is a document which is commonly written in HTML and translated by a web browser. It can be of the static or dynamic type. With the help of HTML only, we can create static web pages.

In essence, HTML is a markup language that empowers developers to craft visually appealing web pages with precise styling. By utilizing a combination of HTML tags, each serving a specific purpose, developers can structure content and create engaging online experiences. Through the utilization of HTML, the web becomes a medium for sharing information and connecting individuals across the globe.

Fig 2.1 (HTML)

Fig 2.2 (CSS)

2.2 CSS

Cascading Style Sheets, commonly referred to as CSS, is a powerful style sheet language that complements markup languages like HTML. Its primary function is to define the visual appearance and formatting of a document. By using CSS in conjunction with HTML, developers have the ability to customize the presentation of web pages and user interfaces, elevating the overall aesthetic and user experience.

CSS can also be applied to XML-based documents, including SVG (Scalable Vector Graphics) and XUL (XML User Interface Language), expanding its utility beyond traditional web pages. In modern web development, CSS is commonly used alongside HTML and JavaScript to build engaging user interfaces for both web and mobile applications.

JavaScript, often abbreviated as "js," is a versatile and lightweight object-oriented programming language extensively utilized for scripting webpages. It serves as an interpreted programming language capable of delivering dynamic interactivity to HTML documents. Initially introduced in 1995 for integrating programs into web pages on the Netscape Navigator browser, JavaScript has since gained widespread adoption across various graphical web browsers.

2.3 JavaScript

JavaScript empowers developers to create modern web applications that offer seamless interactivity without the need for page reloads. It enables the construction of engaging user interfaces and facilitates real-time interactions between users and web content. By leveraging JavaScript, websites can provide a multitude of interactive features, enhancing user experience and simplifying complex tasks.

As a foundational technology, JavaScript plays a pivotal role in enabling interactivity and functionality in traditional websites. It enables developers to incorporate interactive forms, responsive design elements, and dynamic content updates. With JavaScript's capabilities, websites can deliver a more immersive and user-friendly experience, facilitating seamless interactions and ensuring simplicity in navigating and engaging with web content.

Some Applications of JavaScript are: -

- Client-side validation
- Displaying pop-ups, dialog boxes, clocks
- Displaying dynamic content

Fig 2.3 (JavaScript)

2.4 Python

Python is a versatile and widely adopted high-level programming language known for its emphasis on code readability. It boasts a design philosophy that promotes clean and visually uncluttered code through the use of significant indentation, following the off-side rule.

As a dynamically typed language, Python offers flexibility and ease of use. It supports multiple programming paradigms, including structured, object-oriented, and functional programming. This versatility allows developers to choose the most suitable approach for their projects. Additionally, Python is equipped with a comprehensive standard library, often referred to as a "batteries included" feature, providing a wide range of pre-built modules and functions to streamline development.

One of Python's distinguishing features is its focus on readability. Unlike many languages that use curly brackets or keywords to define code blocks, Python relies on whitespace indentation. This indentation-based approach enhances code clarity and facilitates the organization of program logic. By using indentation to denote block structure, Python code becomes more visually appealing and less cluttered.

Some Advantages of using Python are: -

- It offers a user-friendly and intuitive programming experience
- It is an interpreted language which means that the code is implemented line by line
- It has an extensive library and packages to work with

- It is dynamically typed
- It is portable in nature

Fig 2.4 (Python)

2.5 YOLO

YOLO (You Only Look Once) is an object detection algorithm that revolutionized the field of computer vision. Unlike traditional methods, YOLO treats object detection as a single regression problem, making it extremely fast. It divides the input image into a grid and predicts bounding boxes, class probabilities, and confidence scores for each grid cell. Each bounding box consists of coordinates (x, y) representing its center, width (w), height (h), and confidence in the prediction. The grid cells responsible for detecting an object are determined based on the bounding box location. It uses a single neural network to simultaneously predict multiple objects, making it a one-shot detection system. It generates predictions at multiple scales to handle objects of different sizes and applies non-maximum suppression to eliminate redundant and overlapping bounding boxes.

YOLO can detect objects across different categories and achieves impressive real-time performance, allowing for applications in autonomous driving, surveillance, and robotics. YOLO continues to be an influential algorithm, inspiring advancements in the field of object detection and serving as a foundation for subsequent architectures.

In our project we have used YOLOv8, which is the latest version of YOLO by Ultralytics. It is a cutting-edge, state-of-the-art (SOTA) model which has new features and improvements for enhanced performance, flexibility, and efficiency. YOLOv8 supports a full range of vision AI tasks, including detection, segmentation, pose estimation, tracking, and classification. This versatility allows users to leverage YOLOv8's capabilities across diverse applications and domains.

Fig 2.5 (YOLO)

2.6 Flask

Flask is a lightweight and flexible web framework written in Python. It provides developers with the required tools and libraries to build web applications efficiently and in no time. Flask follows the model-view-controller (MVC) architectural pattern, allowing for modular and scalable development. It offers a simple and intuitive interface, making it an excellent choice for beginners and experienced developers alike.

Flask encourages the use of Python's built-in capabilities, providing a minimalistic yet powerful framework for web development. It supports routing, template rendering, and request handling, enabling the creation of dynamic and interactive web pages. Flask also offers extensions for various functionalities, such as database integration, authentication, and form validation. Its modular nature allows developers to choose and incorporate only the necessary components, resulting in lightweight and optimized applications.

Flask's extensive documentation and vibrant community make it easy to find resources, tutorials, and support for any project. It is widely used in the development of web applications, APIs, and prototypes due to its simplicity, flexibility, and scalability. Flask's popularity and versatility have made it a preferred choice among developers for building web-based solutions.

Fig 2.6 (Flask)

2.7 OpenCV

OpenCV (Open-Source Computer Vision Library) is a powerful and popular open-source computer vision and image processing library. It offers a comprehensive set of functions and algorithms that enable developers to perform various tasks related to computer vision, such as image and video processing, object detection and tracking, feature extraction, and more.

OpenCV has an extensive collection of pre-built functions, which simplifies the development process and saves time. It offers a wide range of image and video manipulation tools, allowing users to perform tasks like filtering, resizing, and transformation with ease. Additionally, OpenCV provides algorithms for advanced operations, such as edge detection, feature matching, and image segmentation, which are fundamental to many computer vision applications.

OpenCV also includes machine learning capabilities, with pre-trained models and algorithms that facilitate tasks like object recognition, face detection, and gesture recognition. This integration of computer vision and machine learning makes OpenCV a powerful tool for developing intelligent applications.

Furthermore, OpenCV is platform-independent and compatible with various operating systems, including Windows, Linux, macOS, and even mobile platforms like Android and iOS. Its cross-platform nature makes it highly versatile and widely adopted in industries such as robotics, augmented reality, medical imaging, and surveillance.

Fig 2.7 (OpenCv)

2.8 NumPy

NumPy (Numerical Python) is a fundamental library for scientific computing in Python. It offers support for efficient array operations and mathematical functions, making it a powerful tool for numerical computations. NumPy introduces a new data structure called the "ndarray" (N-dimensional array), which allows for efficient manipulation of large datasets.

With NumPy, complex mathematical operations can be performed on arrays without any difficulty. It offers a wide range of mathematical functions, including algebraic operations, transformations and random number generation. NumPy also integrates easily with other scientific libraries, such as SciPy and Matplotlib, enabling comprehensive data analysis and visualization. The library's multidimensional arrays and broadcasting capabilities simplify data handling and enable efficient vectorized operations.

Its popularity among the scientific and data analysis communities has led to extensive community support, active development, and a vast ecosystem of third-party libraries built on top of NumPy. Overall, NumPy plays a crucial role in scientific computing, enabling efficient numerical operations and facilitating the implementation of advanced algorithms in Python.

Fig 2.8 (NumPy)

2.9 Threading & Queue

Threading in Python enables concurrent execution of multiple threads within a process, improving performance and responsiveness. It provides classes and functions for creating, controlling, and managing threads, allowing for parallelism in tasks.

The queue module provides thread-safe data structures for inter-thread communication. It offers implementations of FIFO queues and LIFO stacks, ensuring safe sharing of data among threads.

Together, threading and the queue module simplify concurrent programming, enabling efficient resource utilization and synchronized communication among threads. However, careful consideration of thread safety is essential to avoid issues like race conditions and data inconsistency.

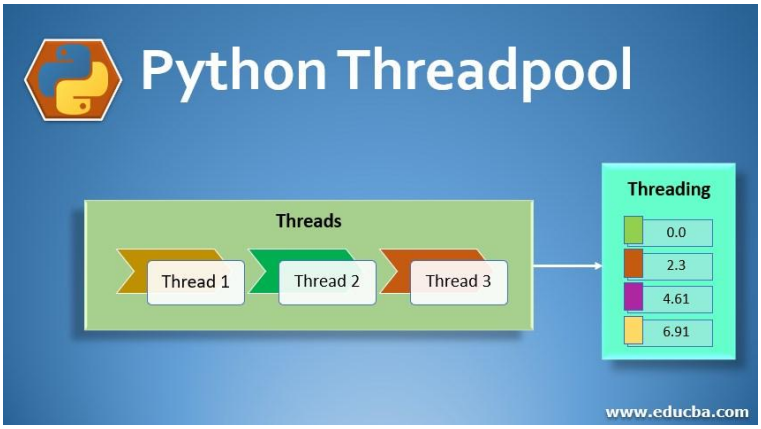


Fig 2.9 (Threading & Queue)

2.10 CORS

Flask_Cors is a Flask extension that helps to handle Cross-Origin Resource Sharing (CORS) in Flask applications. It is a security feature implemented by web browsers that restricts web pages from making requests to a different domain than the one that served the web page. This restriction is in place to prevent malicious websites from accessing sensitive data on other websites.

Flask_Cors is a useful tool for Flask developers who want to enable cross-domain requests in their applications while maintaining security measures. It simplifies the process of handling CORS-related issues, which can be time-consuming and complicated to implement manually.

Fig 2.10 (CORS)

2.11 Indoor Crowd Management

Indoor crowd management refers to the systematic process of controlling and regulating the movement and behavior of individuals within a building or indoor space. This management system ensures that individuals can safely navigate through the building and exit in an orderly manner during an emergency, such as a fire or natural disaster. The basic idea behind indoor crowd management is to minimize the risk of harm or injury to people in a crowded indoor setting.

Effective indoor crowd management involves careful planning, clear communication, and the implementation of appropriate safety measures. This may include the use of signage, designated entry and exit points, security personnel, and emergency evacuation procedures. The goal is to ensure that people can move safely and efficiently through the building, without causing congestion or panic.

In the event of an emergency, such as a fire, indoor crowd management systems are designed to facilitate a quick and safe evacuation of the building. This may involve the use of alarms, emergency lighting, and designated evacuation routes. The effectiveness of indoor crowd management is crucial in preventing injuries, loss of life, and property damage during an emergency situation.

Drawbacks in the current system are because of the use of sign boards put in place for exit as:

- Limited visibility: In the event of a power outage or low visibility due to smoke, sign boards may not be visible, making it difficult for people to locate emergency exits.
- Misleading signage: If the signage is not accurate or up-to-date, it can lead people to the wrong exit, potentially putting them in danger.
- Language barriers: If the sign boards are not in a language that is understood by all people in the building, it can create confusion and delay in evacuating.
- Cognitive overload: In high-stress situations, people may not be able to read and process information on sign boards quickly enough to make informed decisions.
- Compliance issues: Sometimes, people may ignore or overlook sign boards, particularly if they are in a hurry or not familiar with the building.

The basic concept behind the working of this project is the use of Cameras to count the number of people at each exit and then check if it is greater than the threshold

assigned. If the count is less than the threshold then the same path would be used as long as crowd count is less than the threshold. The moment it crosses the threshold, the current exit is no longer recommended and another exit is selected. The major function of this framework is to make sure that people leave the building as soon as possible without getting stuck in any type of chaos. To facilitate this, color coding has been done such that everyone can understand with the use of three basic colors: RED, GREEN and WHITE. The Red color signifies that the exit is no longer recommended and it should not be used. Green color indicates that the exit is safe to use and the White color is used to denote the exits which are not generally used and it is labeled as an IDLE exit.

Advantages of using the suggested frameworks are as follows:

- Accurate crowd count: Cameras can accurately count the number of people at a particular exit and can provide alerts when needed to prevent stampedes.
- Real-time monitoring: The use of cameras can provide real-time monitoring of the situation, allowing emergency responders to quickly identify areas of congestion or blockage and take appropriate action.
- Improved response time: By having a real-time view of the situation, emergency responders can respond more quickly and efficiently, potentially saving lives.
- Objective data: The use of cameras provides objective data that can be used to inform decision-making and improve crowd management strategies.

Overall, using cameras to count and manage crowds during an emergency can help ensure the safety of individuals in a crowded indoor setting. It allows for a quick and efficient response to emergencies, improving the chances of a safe and orderly evacuation.

Fig 2.11 (Indoor Crowd)

Fig 2.12 (Exit)

Chapter 3

Requirement Specifications

3.1 Project Planning: -

- **Objective**: - Developing an Efficient Indoor Crowd Management System using YOLO (object detection) and Flask & Vanilla JavaScript (web framework).

- **Scope**: - The scope of this project is to monitor the flow of people inside the building and identify potential overcrowding or congestion and directing them to the less congested path in case of an emergency or to avoid stampede .

- **Requirements**: - The requirements of the system are :
  - Accurate real time Crowd Detection
  - Visual representation of Crowd data
  - User friendly and responsive web interface

- **Technology Selection**: - We have used YOLOv8 for object detection and HTML, CSS, JavaScript for web development and Flask for integration.

- **Methodology: -**
  - Data Collection: Capturing real-time footage using webcam and mobile camera.
  - Model: Using a pre-trained YOLOv8 model which works on Coco Dataset.
  - Real-time Crowd Detection: Implement functions to read frames from cameras, apply the YOLO model for crowd detection, draw bounding boxes, and calculate crowd count.

- Integration with Flask: Integrate the model with Flask by setting up routes and endpoints to handle video streams and perform real-time crowd detection.
- User Interface: Develop a user interface using HTML templates and Flask's render_template function to display video streams and crowd count data and connect this page to the home page.
- Multithreading: Employ threading to run crowd count generation and frame generation concurrently, enhancing system performance.
- CORS: enabling cross-domain requests in their applications while maintaining security measures.

❖ **Milestones and Timeline: -**

- Milestone 1 - Data collection and Model finalization (Week 1-2)
- Milestone 2 - Model Setup and Webcam setup (Week 3-4)
- Milestone 3 - Real-time Crowd Detection and Testing (Week 4-8)
- Milestone 4 - Flask Integration and UI Development (Week 9-10)
- Milestone 5 - Testing & Finalization (Week 11-12)

❖ **Resource Allocation: -**

- Hardware Resources: Webcam, Mobile camera and Laptop.
- Software Resources: Python, Flask, OpenCV, NumPy, Threading and Queue module, YOLO, and Ultralytics library.

❖ **Risk and Mitigation: -**

- Risk: System performance may be affected by hardware limitations (No GPU availability).
- Mitigation: Perform optimizations to ensure the system operates within the available resources.

❖ **Testing: -**

Testing the model by checking whether the real-time crowd detection model is accurate or not for both small and large crowds.

❖ **Project Management: -**

In order to effectively manage our project, we adopted an Agile Project Management approach. We broke down the development process into iterative sprints, conducting weekly meetings with our Project Guide to review our progress, address challenges, and adapt the project plan as needed. Regular communication among team members was maintained to facilitate coordination and mitigate any potential risks or delays.

3.2 Project Specification/SRS:-

Functional Requirements: -

- Video Feed Capture: The system should be capable of capturing live video feeds from multiple cameras placed in the indoor environment.
- Object Detection: The system should perform real-time object detection and specifically identify individuals within the captured video frames.
- Crowd Counting: The system should accurately count the number of individuals present in the indoor space at any given time.
- Live Video Display: The system should display the live video feed with overlaid bounding boxes around detected individuals for visual confirmation.
- Threshold-based Alerts: The system should generate real-time alerts or notifications when the crowd count exceeds a predefined threshold.
- User Interface: The system should provide a user interface that allows users to configure and adjust the crowd count threshold and other relevant parameters.

Non-functional Requirements: -

- Performance: The system should provide real-time processing and detection of individuals within the captured video frames to ensure timely crowd monitoring.
- Scalability: The system should be capable of handling a significant number of cameras and processing multiple video feeds simultaneously without compromising performance.
- Availability: The system should have a high level of availability, ensuring that it operates continuously without significant downtime or interruptions.
- Usability: The system should have a smooth and user-friendly interface. The system should be accompanied by comprehensive documentation, including guides, to assist users in understanding the system effectively.
- Compatibility: The system should be compatible with various types of cameras commonly used for indoor surveillance, ensuring seamless integration and video feed capture and the system should be compatible with different operating systems and web browsers to accommodate a wide range of users.

User Requirements: -

- Crowd Monitoring: The system should be able to accurately detect and count individuals within indoor environments, providing real-time crowd monitoring capabilities.
- Real-time Alerts: Users require the system to promptly notify them when the crowd count exceeds predefined thresholds, and should redirect the crowd.

- User-Friendly Interface: The system should provide user-friendly controls to enhance user experience.
- Scalability and Flexibility: The system should be scalable to accommodate multiple locations, different camera configurations, and growing crowd sizes without compromising performance.

System Requirements: -

- Hardware Requirements:
  - The system should be compatible with a webcam capable of capturing high-resolution video.
  - Sufficient processing power and memory to handle real-time video processing and crowd detection algorithms.
  - Adequate storage capacity should be provided to store video data for analysis and future reference.
- Software Requirements:
  - The system should utilize appropriate software frameworks and libraries for efficient video processing, crowd detection, and analytics.
  - Integration with flask and correct file structure is important.
- Network Requirements:
  - The system should support network connectivity to receive video streams from multiple cameras and transmit processed data for analysis.
  - Sufficient bandwidth and network infrastructure should be available to handle the data transmission requirements.
- Dependency Requirements:
  - The system may rely on third-party components, such as machine learning libraries or cloud-based services, for advanced video analytics.
  - Compatibility with specific versions or dependencies should be considered to ensure seamless integration and functionality.

Use Cases: -

- Scenario: Real-time Crowd Monitoring
  - **User Interaction:** The user selects a specific camera feed for real-time monitoring.
  - **System Response:** The system displays the live video feed from the selected camera and provides real-time updates and visual indicators like color green and red to show the crowd status and whether it has exceeded the threshold and is safe for more crowds to access it or not.

- Scenario: Crowd Threshold Exceeded
  - **User Interaction:** The user sets a threshold value for the maximum allowable crowd count.
  - **System Response:** When the crowd count surpasses the threshold, the system triggers an alert and changes the colour to red to indicate the overcrowding situation. It redirects the user to the other staircase for crowd management.

- Data Requirements: -

  - Data Structures: The system is able to process both image and video data from the webcams installed in the indoor space. The data is represented in a format that can be easily analysed and processed by the system.
  - Data Processing Requirements: The system is able to process data in real-time, enabling quick detection and response to changes in crowd density. The system is also able to process data in batches for offline analysis and reporting.
  - Data Privacy Requirements: The system adheres to data privacy regulations and protects the privacy of individuals captured in the video and image data.

System Design Constraints: -

  - Hardware Limitations: The performance of the system depends on the hardware used, such as the processing power of the CPU and the speed of the storage device. Therefore, the system design should consider the hardware limitations and ensure that the system can process and analyse the data in real-time.
  - Scalability: The system may need to handle a large volume of data from multiple cameras in different locations. Therefore, the system design should consider scalability and ensure that the system can handle a large amount of data and perform well under heavy loads.

Assumptions and Dependencies: -

  - Camera Placement: The system assumes that the webcams are placed in an optimal position to capture a clear and comprehensive view of the indoor space.

- Image Quality: The system assumes that the webcams capture high-quality images that can be easily processed by the computer vision algorithms
- Webcam Hardware: The system relies on webcams to capture the footage, and therefore, the system's performance is dependent on the quality and specifications of the webcams.
- Computer Vision Libraries: The system relies on computer vision libraries, such as OpenCV, to analyse the images captured by the webcams and detect people and estimate the crowd density.

Verification and Validation: -

- Unit Testing: performing unit testing to ensure that each module of the system works as intended and meets the requirements.
- Integration Testing: Integrating the modules and testing them together to ensure that they work together seamlessly.
- System Testing: Testing the system as a whole to ensure that it meets the requirements and works as intended.
- Reviews and Inspections: The system's design, code, and documentation are reviewed and inspected by a team of experts to identify any defects, errors, or issues that may affect the system's quality and reliability.
- Validation: The system's accuracy and performance will be validated against real-world data to ensure that it meets the expected outcomes.

Documentation and Deliverables: -

- Requirements Document: This document outlines the requirements for the indoor crowd management system model, including the features and functionalities that it should support
- Design Document: This document describes the system's architecture, design, and implementation, including the hardware and software components, algorithms, and data structures used.
- System Documentation: This document provides a detailed overview of the system's components, workflows, and processes, including data flow diagrams, system diagrams, and technical specifications.
- Test Reports: These reports detail the testing results of the indoor crowd management system model, including the test scenarios, test cases, and test results.

3.3 System Design

3.3.1 Design Constraints: -

- Hardware Limitations: The system's design should take into account the hardware constraints- processing power and memory capacity of the devices used for crowd monitoring. Efficient algorithms and optimization techniques should be employed to ensure real-time performance within the hardware limitations.

- Network Bandwidth: The system should consider the limitations of the network bandwidth when transmitting live video feeds and data from multiple cameras to the central processing unit or monitoring interface.

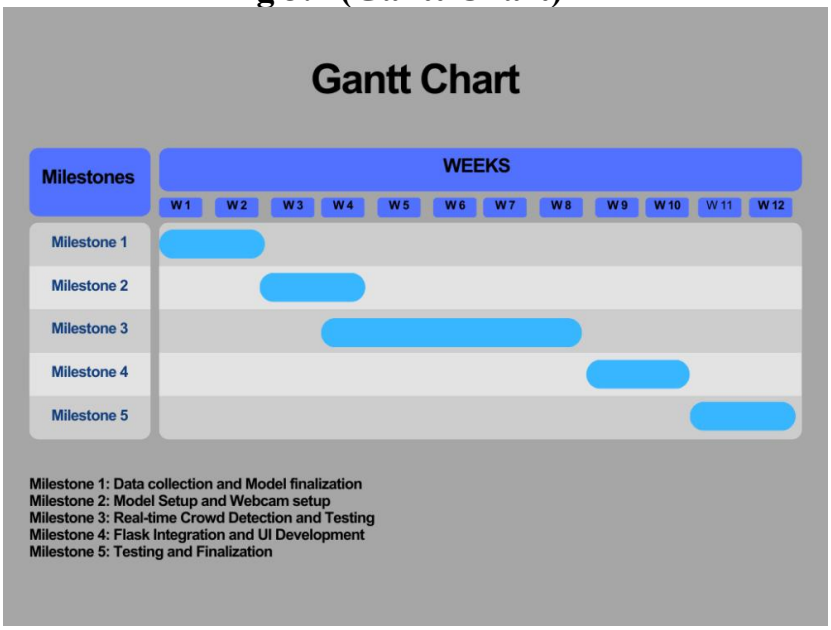
- Scalability: The design should be scalable to accommodate an increasing number of cameras and locations for crowd monitoring without any compromise in its performance and response time.

3.3.2 System Architecture: -

We have used three diagrammatic representations to explain the system architecture of our project.

- Gantt Chart - It provides a clear overview of our project timeline, tasks, dependencies, and milestones.

Fig 3.1 (Gantt Chart)



- Product Flow Diagram (PFD) - It visually depicts how inputs are transformed into outputs, the sequence and the interactions of various components of the project.

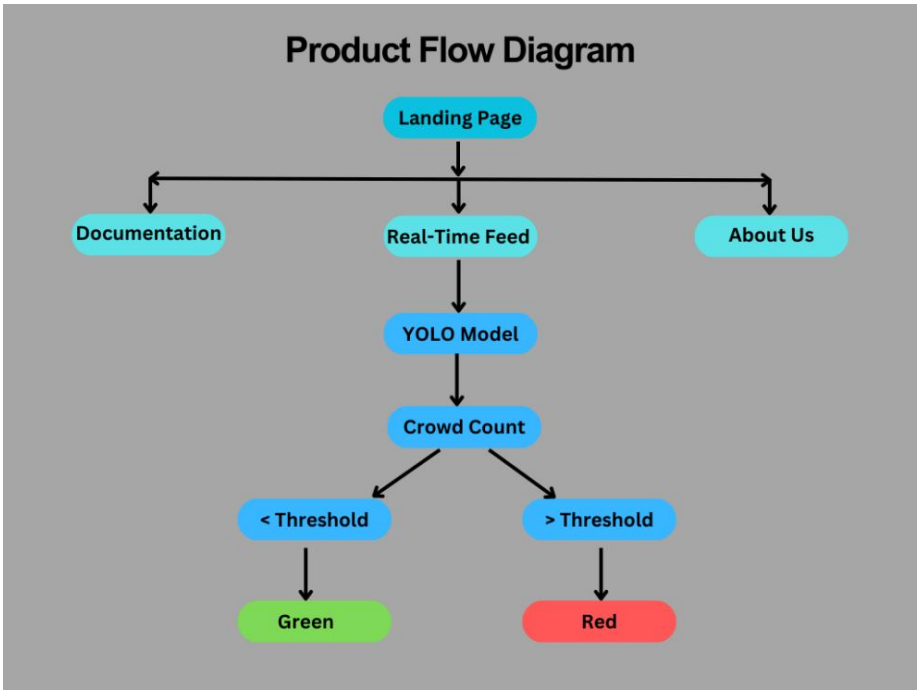


Fig 3.2 (Product Flow Diagram)

- Unified Modeling Language (UML) diagram- It provides a set of graphical notations for representing different aspects of a system, such as its structure, behavior, and interactions. The UML diagrams that we have included in our project report are Class Diagram and Use-case diagram.

  - ♦ Use-Case Diagram - it illustrates the interactions between actors and a system, capturing the various use cases or functionalities of the system.

  Here, 3 actors - Administrator, Visitor and Authorities and, 2 Primary cases - Check density and Manage Crowd are used.

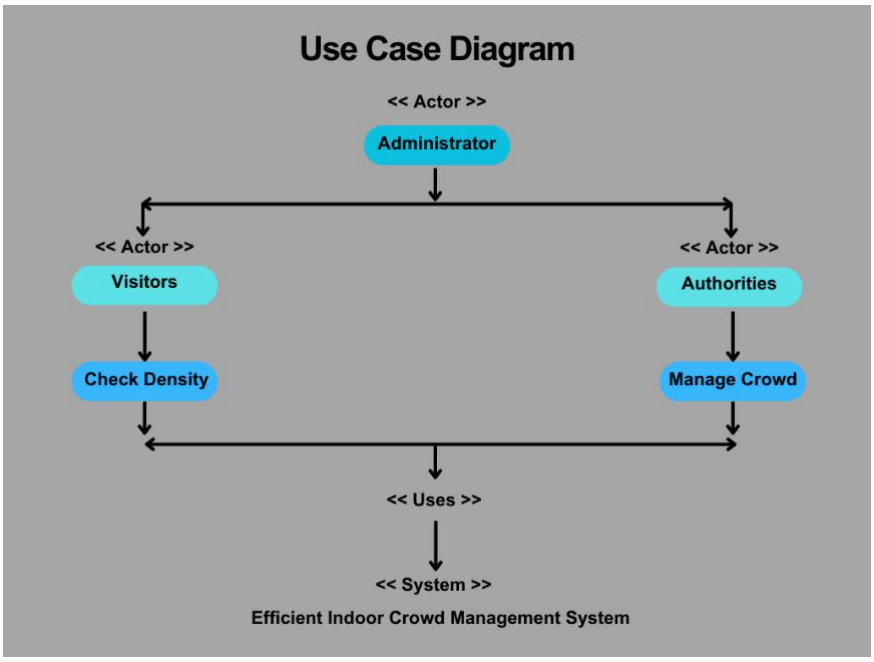


Fig 3.3 (Use Case Diagram)

- Class Diagram - it provides a structural view of the system by illustrating the classes, their attributes, and the relationships between them. Here,
  - ♦ Crowd Analytics - This class is responsible for analyzing the crowd density and generating alerts when necessary.
  - ♦ Visitors - This class represents different users who interact with the system.
  - ♦ Detection and alert - This class represents the different types of events that can be generated by the system
  - ♦ Configuration - This class is responsible for storing the generated alerts according to the threshold provided.

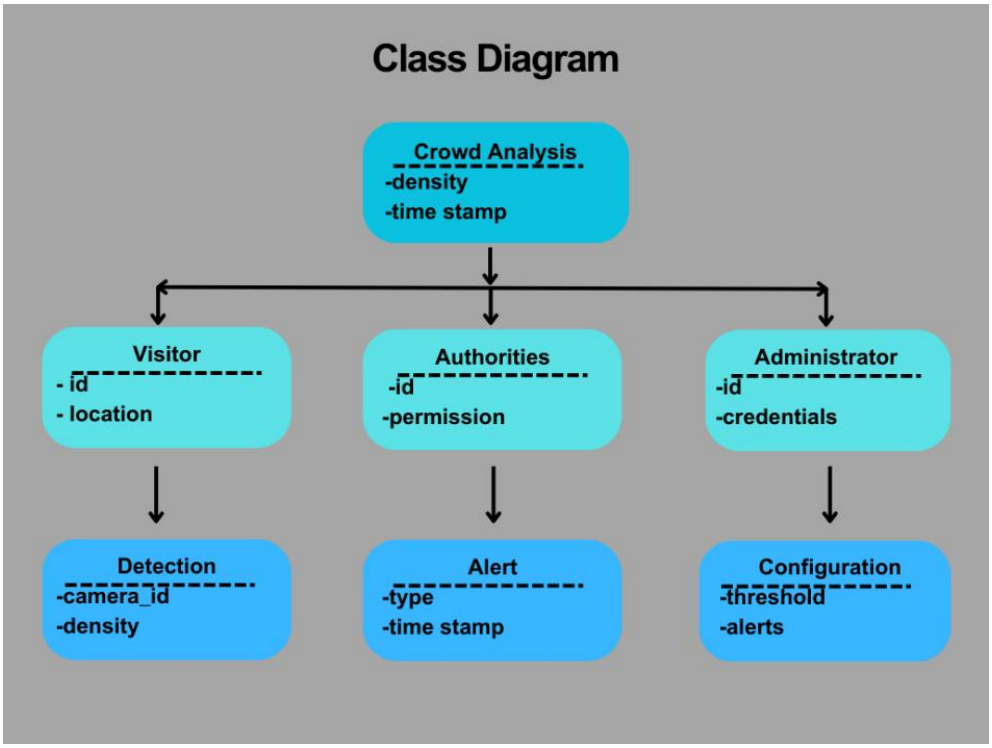


Fig 3.4 (Class Diagram)

Chapter 4

Implementation

4.1 Methodology: -

- Data Collection: The project involved capturing real-time video feeds from both webcam and mobile cam. The cameras were connected to the system using OpenCV library, which provided access to video streams for further processing.

- Crowd Detection and Tracking: The YOLO (You Only Look Once) algorithm from the Ultralytics library was utilized for crowd detection and tracking. This algorithm was implemented using Python and integrated into the system using Flask.

- Crowd Counting: The system utilized the detected crowd bounding boxes from YOLO to calculate the crowd count in each frame. A counting mechanism was employed to determine the number of persons present in the field of view of each camera. The count was then aggregated to provide an overall crowd count for the indoor environment.

- UI Development: To provide real-time crowd information to the management team, a web-based user interface was developed using Flask. The interface displayed the live video feed from each camera along with the corresponding crowd count. The interface included a landing page and a dashboard to monitor and analyze crowd patterns over time.

- Performance Optimization: To ensure efficient processing of video feeds and real-time crowd counting, multithreading was implemented using Python's

threading module. This allowed simultaneous execution of video capture, crowd detection, and crowd counting tasks, optimizing the system's performance.

- Validation and Testing: The system was extensively tested in a controlled indoor environment to validate the accuracy and reliability of crowd detection and counting. Test scenarios with varying crowd densities and movements were conducted to evaluate the system's performance.

4.2 Testing Plan :-

Test ID	Test Case Title	Test Condition	System Behavior	Expected Result
T01	Crowd Detection Accuracy	Varying crowd densities and movements	Accurate detection and tracking	Minimum False positives and correct bounding boxes
T02	Crowd Counting Accuracy	crowd densities ranging from low to high	Accurate count being displayed	crowd count displayed and the actual count should have an acceptable margin of error
T03	Real-Time Video Feed	Live video feed from both cameras	Video feeds should be displayed without any delay or buffering	Smooth Real-time video display
T04	Optimized Performance	Simultaneous processing of real-time video and crowd count	System should handle processing load efficiently	Consistent performance without lag or slowdown
T05	User Interface functionality	Interaction with the web-based UI	UI should be responsive and smooth	Able to view dynamic crowd count and real-time video feed

4.3 Result Analysis / Screenshots

Landing page/Home page: -

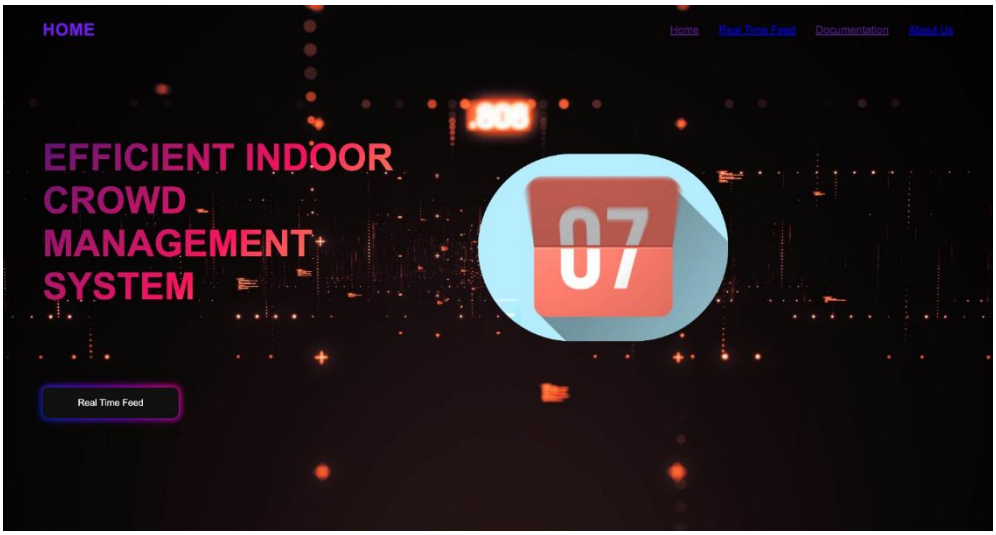


Fig 4.1 (Landing Page)

The Landing page connects to 4 pages: -

- Real-time Feed - Real time crowd counting using webcams and visual color cues to redirect the crowd above a threshold value (Here, it is 3)
  This is the idle case where Primary exit (1) is Green and should be used and 2 is idle.

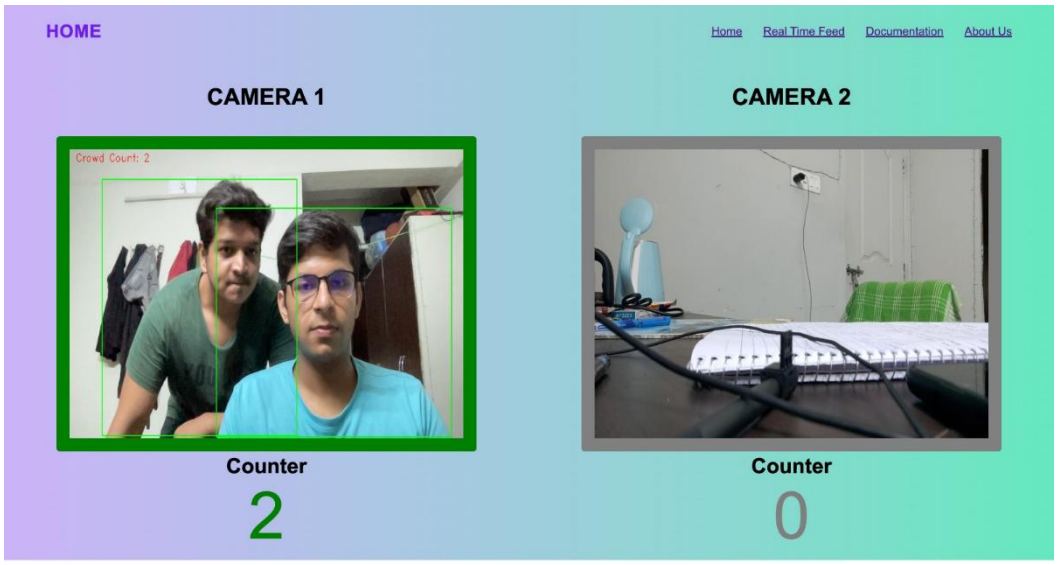


Fig 4.2 (Real-time Feed)

This is the case where crowd count in Primary exit (1) crosses the threshold and the crowd should be redirected to 2, so the color of Exit 1 changes to Red and 2 becomes Green.

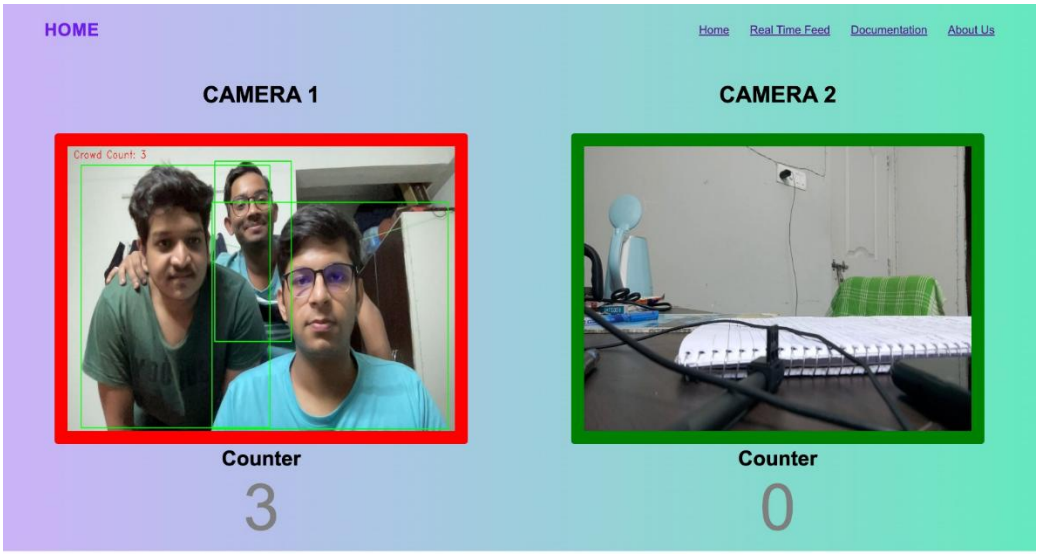


Fig 4.3 (ALERT CASE)

Here, the crowd has been successfully redirected, so color changes back to Green.

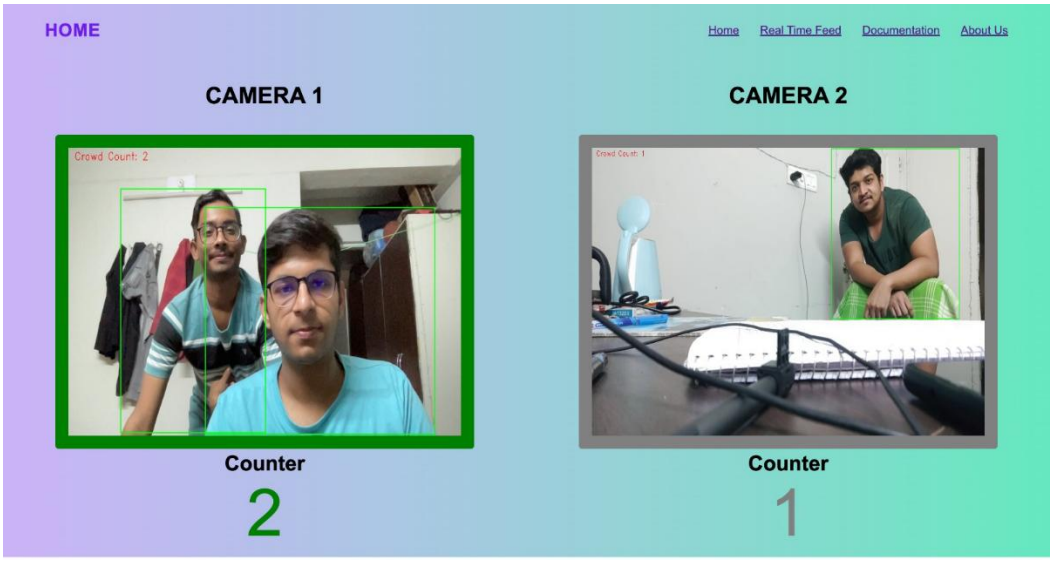


Fig 4.4 (REDIRECT CASE)

- Documentation - It shows a brief description of our project

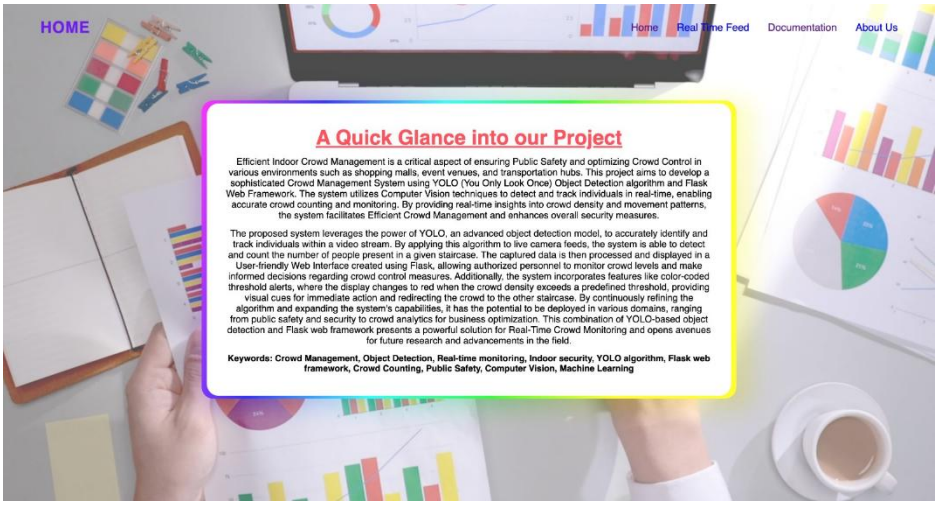


Fig 4.5 (Documentation)

- About Us - This page provides the contact handles of all the team members and the Project Guide

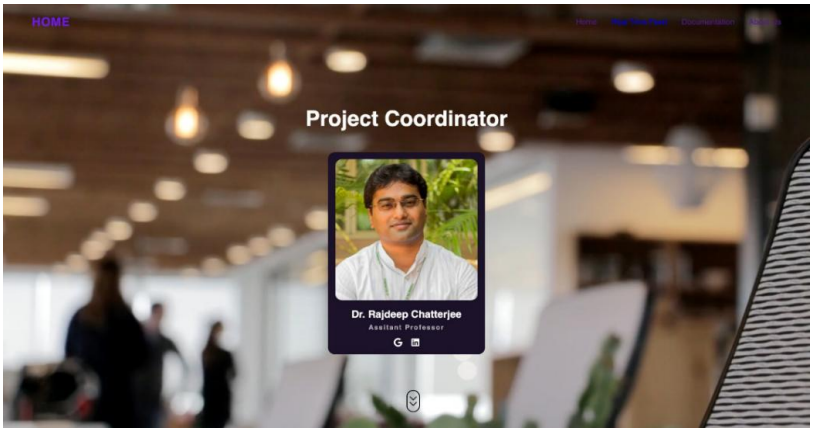


Fig 4.6 (About Us 1)

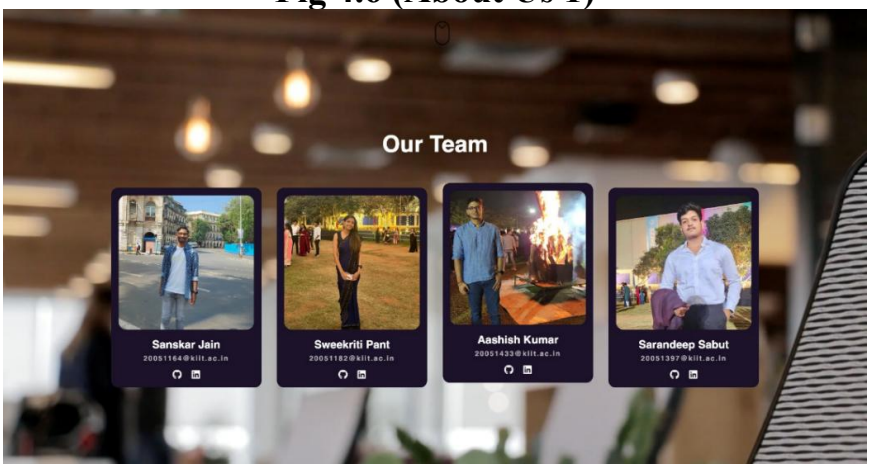


Fig 4.7 (About Us 2)

4.4 Quality Assurance

Quality assurance for this project was ensured through a rigorous testing plan and various verification processes. The Project Guide, under whose guidance we are doing this project, reviewed and evaluated it against predefined criteria and guidelines to ensure adherence to quality standards. Regular feedback and revisions were incorporated to address any identified areas of improvement, resulting in a high-quality final product.

Chapter 5

Standards Adopted

5.1 Design Standards: -

For our project, the following design standards were followed:

- Use of UML Diagrams: UML diagrams such as use case diagrams and class diagrams were used to represent the system's structure, behavior, and interactions.
- Modularity and Component-based Design: The system was designed with a modular approach, breaking it down into smaller components with well-defined interfaces promoting code reusability, maintainability, and flexibility.
- Code Documentation: Proper code documentation practices using standard documentation tools and techniques such as comments, inline documentation, and documenting function signatures was done.
- Code Styling and Naming Conventions: Consistent coding style and naming conventions were followed throughout the project.
- Performance Optimization: Performance optimization techniques such as use of efficient algorithms and caching mechanisms was done to enhance the system's speed, responsiveness, and scalability.
- Usability and User Experience (UX): A smooth and responsive user interface (UI) was designed with usability and UX principles in mind to enhance user satisfaction and ease of use.
- Testing and Quality Assurance: Established testing methodologies were followed to ensure the reliability, functionality, and correctness of the system.

5.2 Coding Standards: -

For our project, the following coding standards can be considered:

- Keep Code Concise: Writing code using as few lines as possible while maintaining readability and clarity.
- Follow Naming Conventions: Using appropriate naming conventions for variables, functions, and classes
- Use Clear Code Segmentation: Blocks of code were organized within the same section into paragraphs or logical groupings. This improves readability and makes it easier to understand the flow and structure of the code.
- Indentation for Control Structures: Use of indentation consistently to clearly mark the beginning and end of control structures such as loops and conditionals. This helps in visually identifying the scope of code within these structures.
- Single-Responsibility Functions: Aimed at keeping functions focused on carrying out a single task or responsibility.
- Consistent Code Formatting: Maintaining consistent code formatting throughout the project.
- Code Testing: Written testable code by incorporating unit testing and integration testing.

5.3 Testing Standards: -

The standard followed for testing and verification of our project work is ISO/IEC 9126 - this standard focuses on software product quality and provides guidelines for quality characteristics and metrics such as functionality, reliability, usability, efficiency, maintainability, and portability.

Chapter 6

Conclusion and Future Scope

6.1 Conclusion

The project on efficient indoor crowd management utilizing YOLO, Flask and JavaScript has achieved significant milestones and demonstrated its potential in addressing the challenges of crowd management in indoor environments. By leveraging advanced computer vision techniques and a user-friendly web interface, the system offers real-time crowd monitoring, accurate crowd counting, and enhanced situational awareness.

Throughout the project, a systematic approach was followed, incorporating rigorous planning, thorough analysis, efficient system design, and meticulous testing. The implementation adhered to established coding standards and best practices, ensuring readability, maintainability, and scalability of the codebase. The project team successfully integrated the YOLO object detection model with Flask, harnessing the power of real-time video processing and crowd analysis.

The testing and verification phase incorporated industry-standard practices, including test case design, test execution, and result validation. By following ISO and IEEE testing standards, the project team ensured the reliability, functionality, and performance of the system. Robust testing scenarios encompassing various crowd densities, lighting conditions, and camera angles were conducted, validating the system's accuracy, responsiveness, and scalability.

The results obtained from extensive testing and evaluation demonstrated the effectiveness of the system in accurately detecting and counting individuals within a crowd. The system's user-friendly interface provided valuable insights into crowd

dynamics, enabling efficient resource allocation, crowd management strategies, and proactive decision-making. The successful implementation of the project showcased its potential to enhance safety, security, and operational efficiency in diverse indoor environments such as airports, stadiums, malls, and public venues.

In conclusion, the project on efficient indoor crowd management has achieved its objectives by leveraging state-of-the-art technologies and adhering to industry best practices. The developed system offers a reliable, scalable, and user-friendly solution for real-time crowd monitoring and management. The project not only contributes to the field of computer vision and crowd analysis but also holds immense potential for practical applications in various industries. Through its successful implementation, the project paves the way for enhanced crowd management strategies, ensuring safer and more efficient environments for individuals in crowded indoor spaces.

6.2 Future Scope

Some of the future scopes for our project are: -

- Exploring the integration of machine learning algorithms like deep reinforcement learning for more intelligent and adaptive crowd management strategies.
- Investigating the feasibility of incorporating social distancing protocols and crowd density estimation techniques for enhanced safety measures.
- Enhancing the system's capabilities by integrating multi-camera setups and leveraging advanced computer vision techniques for comprehensive crowd monitoring.
- Implementing real-time data analytics and visualization tools to provide actionable insights for efficient decision-making and resource allocation.
- Extending the project to outdoor environments and large-scale events, addressing the unique challenges and requirements associated with open spaces and diverse crowd dynamics.

References

- A Mobile Based Crowd Management System Wafaa M. Shalash,, Aliaa Al Hazimi, Basma Al Zahrani
- Object Detection through Modified YOLO Neural Network
- Tanvir Ahmad ,Yinglong Ma ,Muhammad Yahya,Belal Ahmad,Shah Nazir ,and Amin ul Haq
- Real-Time Object Detection with Yolo Geethapriya. S, N. Duraimurugan, S.P. Chokkalingam
- Analytical Study on Object Detection using Yolo Algorithm Dawn Wilson, Dr. Manusankar C , Dr. Prathibha P H
- The IEEE website available: http://www.ieee.org/
- The Javatpoint website available: https://www.javatpoint.com/

EFFICIENT INDOOR CROWD MANAGEMENT SYSTEM

SANSKAR JAIN
20051164

Abstract: The Aim of our project is to develop an Efficient Indoor Crowd Management System using YOLO and Flask. The Objective is to accurately monitor and predict crowd density in real-time, enabling timely interventions and ensuring the safety and comfort of visitors using color-coded alerts. By implementing this system, we aim to enhance crowd management strategies and create a safer and more comfortable environment in public spaces.

Individual contribution and findings: As a member of the project group, my primary responsibility was to work on the frontend development of our indoor crowd management system. This involved creating the user interface, implementing color coding using JavaScript, and ensuring seamless navigation between different pages.

In terms of planning, I collaborated closely with the team to understand the requirements and design a user-friendly interface. I focused on creating an intuitive layout that would enable users to easily visualize crowd density and receive real-time updates. Additionally, I strategized the implementation of color coding to highlight different crowd levels and convey critical information at a glance.

One of the notable findings during this process was the impact of visual cues and color coding on user perception and engagement. By employing effective design principles and JavaScript interactivity, I observed how users could quickly grasp crowd information and make informed decisions. The ability to connect the frontend pages together facilitated a cohesive user journey, enabling easy access to various features and functionalities.

Working on the frontend and color coding aspect provided me with a deeper understanding of user interface design, JavaScript programming, and the importance of creating an intuitive and responsive system. Through this experience, I honed my skills in frontend development and gained insights into effectively connecting different components of a web application.

Overall, this project has enhanced my proficiency in frontend technologies, user experience design, and JavaScript programming. The opportunity to contribute to the development of the user interface, implement color coding using JavaScript, and connect all the pages together has allowed me to acquire valuable practical knowledge and strengthen my abilities as a frontend developer.

Individual contribution to project report preparation: Contributed to Section 2.1, 2.2, 2.3, 3.1, 3.3.2, 5.2 and References in the Project Report Preparation.

Individual contribution for project presentation and demonstration:
Contributed in making Technology Used and Designing of the presentation and demonstrating the Frontend of the project.

Full Signature of Supervisor: Full signature of the student:

....

EFFICIENT INDOOR CROWD MANAGEMENT SYSTEM

SWEEKRIT PANT
20051182

Abstract: The Aim of our project is to develop an Efficient Indoor Crowd Management System using YOLO and Flask. The Objective is to accurately monitor and predict crowd density in real-time, enabling timely interventions and ensuring the safety and comfort of visitors using color-coded alerts. By implementing this system, we aim to enhance crowd management strategies and create a safer and more comfortable environment in public spaces.

Individual contribution and findings: As a member of the project group, my role centered around implementing the YOLO (You Only Look Once) model and resolving the CORS (Cross-Origin Resource Sharing) issue within our indoor crowd management system. Throughout this process, I gained invaluable experience and made significant contributions to the project.

In terms of planning, I thoroughly researched the YOLO model and its suitability for our specific requirements. I collaborated closely with team members to gather relevant crowd data, use the pretrained model, and fine-tune its performance and efficiency. Additionally, I worked alongside the front-end developer to establish seamless communication between the model and frontend components.

One of the notable technical findings during the implementation was the effectiveness of the YOLO model in accurately detecting and tracking individuals. Witnessing the model's ability to estimate crowd density in real-time was a remarkable experience. Resolving the CORS issue further added to my technical expertise, as I implemented secure configurations and ensured smooth data transfer between the various system modules.

Overall, this project provided me with hands-on experience in working with advanced computer vision techniques and overcoming technical challenges. The opportunity to integrate the YOLOv8 model and address the CORS issue broadened my understanding of system integration and security considerations. This experience has reinforced my skills in developing efficient and reliable solutions while contributing to the successful implementation of our indoor crowd management system.

Individual contribution to project report preparation: Contributed to Abstract, Section 2.6, 2.7, 2.8, 4.1, 4.3, 6.1 in the Project Report Preparation.

Individual contribution for project presentation and demonstration:

Contributed in making Implementation and Future Scope of the presentation and demonstrating the Yolo Model detection of the project.

Full Signature of Supervisor: Full signature of the student:

....

EFFICIENT INDOOR CROWD MANAGEMENT SYSTEM

B. SARANDEEP SABUT
20051397

Abstract: The Aim of our project is to develop an Efficient Indoor Crowd Management System using YOLO and Flask. The Objective is to accurately monitor and predict crowd density in real-time, enabling timely interventions and ensuring the safety and comfort of visitors using color-coded alerts. By implementing this system, we aim to enhance crowd management strategies and create a safer and more comfortable environment in public spaces.

Individual contribution and findings: As a member of the project group, my main responsibility was to work on the integration of the YOLO model with Flask and connect it with the frontend of our indoor crowd management system. This involved handling requests and responses, and ensuring the seamless flow of data between the frontend and the backend.

In terms of planning, I collaborated closely with the team to understand the requirements and design an efficient integration strategy. I focused on creating robust APIs that could handle real-time webcam inputs from the frontend, process them using the model, and return the crowd density predictions. I also coordinated with the frontend developer to establish a smooth connection between the frontend and the Flask backend. During the implementation phase, I gained hands-on experience in backend web development using Flask.

One of the significant findings during this process was the importance of efficient communication between the frontend and the backend. By establishing a seamless connection, I observed how the YOLO model's crowd density predictions could be accurately integrated into the frontend UI, providing real-time insights to users. This integration played a crucial role in delivering a comprehensive indoor crowd management solution.

Overall, this project has enhanced my proficiency in backend web development, API integration, and the integration of machine learning models into web applications. My skills in Python programming got enhanced, I understood Flask framework utilization, and the handling of data transfer between different components of a web system.
The opportunity to contribute to the Flask integration of the YOLO model and connect it with the frontend has allowed me to acquire valuable practical knowledge and skills.

Individual contribution to project report preparation: Contributed to Section 1, 3.2, 3.3.2, 5.2, 5.3 and 6.2 in the Project Report Preparation.

Individual contribution for project presentation and demonstration:
Contributed in making Introduction of the presentation and demonstrating the Flask integration of the Object detection model of the project.

Full Signature of Supervisor: Full signature of the student:

....

EFFICIENT INDOOR CROWD MANAGEMENT SYSTEM

AASHISH KUMAR
20051433

Abstract: The Aim of our project is to develop an Efficient Indoor Crowd Management System using YOLO and Flask. The Objective is to accurately monitor and predict crowd density in real-time, enabling timely interventions and ensuring the safety and comfort of visitors using color-coded alerts. By implementing this system, we aim to enhance crowd management strategies and create a safer and more comfortable environment in public spaces.

Individual contribution and findings: In the project group, my specific role involved working on the implementation of the YOLO (You Only Look Once) model and handling the threading and queuing aspects within our indoor crowd management system.

Regarding planning, I conducted in-depth research on the model and its suitability for our crowd management requirements. I collaborated closely with team members to analyze the system architecture and devise an efficient threading and queuing mechanism to ensure real-time processing and responsiveness of the model.

One of the noteworthy technical findings during the implementation was the effectiveness of the YOLOv8 model in accurately detecting and tracking individuals within crowded spaces. Witnessing the model's capabilities in estimating crowd density in real-time was an exciting experience. Additionally, exploring threading and queuing techniques to optimize the model's performance and enhance system scalability provided valuable insights.

Working on the threading and queuing of the model allowed me to delve into the intricacies of concurrent programming and resource management. Implementing efficient thread management

and task queuing strategies enhanced the system's responsiveness and throughput, ensuring smooth integration of the model into our indoor crowd management system.

Overall, this project enriched my understanding of advanced computer vision techniques, concurrent programming, and system optimization. The opportunity to work on the YOLOv8 model and tackle the challenges related to threading and queuing has strengthened my technical skills and problem-solving abilities. By contributing to the successful implementation of our indoor crowd management system, I have gained valuable experience in developing robust and scalable solutions in complex environments.

Individual contribution to project report preparation: Contributed to Section 2.4, 2.5, 2.11, 3.2, 4.2, 4.4 and 5.2 in the Project Report Preparation.

Individual contribution for project presentation and demonstration:
Contributed in making Working and Implementation of the presentation and demonstrating the Yolo Model detection part of the project.

Full Signature of Supervisor: Full signature of the student:

....

PLAGIARISM REPORT

Efficient Indoor Crowd Management System

ORIGINALITY REPORT

9%	5%	2%	6%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	www.coursehero.com Internet Source	2%
2	Submitted to Miami Dade College Student Paper	1%
3	Submitted to University of Western Sydney Student Paper	1%
4	www.ijert.org Internet Source	<1%
5	Submitted to Queen Mary and Westfield College Student Paper	<1%
6	aceieduonline.wordpress.com Internet Source	<1%
7	docplayer.net Internet Source	<1%
8	recerc.eu Internet Source	<1%
9	Submitted to University of Central Florida Student Paper	<1%

10	Submitted to University of Hertfordshire Student Paper	<1%
11	"Computer Vision", Springer Science and Business Media LLC, 2017 Publication	<1%
12	Submitted to New College Swindon Student Paper	<1%
13	etd.repository.ugm.ac.id Internet Source	<1%
14	Submitted to University of Huddersfield Student Paper	<1%
15	www.projectpro.io Internet Source	<1%
16	dspace.daffodilvarsity.edu.bd:8080 Internet Source	<1%
17	Submitted to National College of Ireland Student Paper	<1%
18	flamerobin.updatestar.com Internet Source	<1%
19	Submitted to Aberystwyth University Student Paper	<1%
20	Boon Ho, Basaran Bahadir Kocer, Mirko Kovac. "Vision based crown loss estimation for individual trees with remote aerial robots",	<1%