JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Live Video Surveillance Camera Using Android

[1]Omkar Dileep Thorat, [2]Abhimanyu Prakash Patil, [3]Sayali Avinash Patil, [4]Raj Dastageer Mulla, [5]Mrs.Hafsa Shoyeb Majgoankar

[1,2,3,4] UG Students, [5]Assistant Professor

[1,2,3,4,5] Department of Computer Science and Engineering

[1,2,3,4,5] Nanasaheb Mahadik College of Engineering, Peth, Maharashtra, India

Abstract : Everyone in world deserves privacy and security, in daily life we come across various situation where we need privacy and security. Surveillance Cameras using smart phones has made it possible for human beings to experience a revolutionary aspect of the Technology. Surveillance Cameras are mostly used to improve security and privacy by using Smartphone as a medium. Home Security cameras, Pet cameras, baby cam eras, Health care cameras are the use cases of Surveillance cameras. This paper presents an interactive application where the user can use smart phone as a Surveillance camera and live streaming can be done by another smart phone where a user can also get instant notifications whenever an object in motion is detected and live streaming done by smartphone can be recorded and stored with the help of cloud storage so that user can get instant recording. This project aims to reuse our smartphones as a security camera (CCTV) with a user-friendly experience that can ultimately lead to a smart camera with low-cost capital.

IndexTerms **- Privacy, Security, Surveillance Camera, Smart Phones, Home Security .**

I. INTRODUCTION

Surveillance cameras are one of the most ubiquitous and recognizable technologies used to watch us as we move about our daily lives. Networks of cameras are installed by government agencies and by local businesses, but the distinction blurs with the development of real-time crime centres that access both public and private video feeds. Camera technology is growing in sophistication: some cameras are capable of 360-degree video or infrared vision. Some models can be equipped with real-time face recognition or license plate recognition software. Since many are also being connected directly to the Internet, the camera networks have also proven easy targets for malicious attackers. Surveillance is the monitoring of behavior, many activities, or information for the purpose of information gathering, influencing, managing or directing. This can include observation from a distance by means of electronic equipment, such as closed-circuit television (CCTV), or interception of electronically transmitted information like Internet traffic. It can also include simple technical methods, such as human intelligence gathering and postal interception. Surveillance is used by citizens for protecting their neighborhoods. And by governments for intelligence gathering - including espionage, prevention of crime, the protection of a process, person, group or object, or the investigation of crime. It is also used by criminal organizations to plan and commit crimes, and by businesses to gather intelligence on criminals, their competitors, suppliers or customers. Religious organizations charged with detecting heresy and heterodoxy may also carry out surveillance. Auditors carry out a form of surveillance. A byproduct of surveillance is that it can unjustifiably violate people's privacy and is often criticized by civil liberties activists. Liberal democracies may have laws that seek to restrict governmental and private use of surveillance, whereas authoritarian governments seldom have any domestic restrictions.

II. LITERATURE REVIEW

The paper titled "Mobile Video Surveillance Using Smartphone Camera for Security Applications" by J. P. Singh, J. Kaur, and M. Singh, explores the use of smartphone cameras for mobile video surveillance in security applications.The authors begin by addressing the increasing demand for video surveillance systems and the limitations of traditional surveillance cameras. They highlight the widespread availability and advanced capabilities of smartphones, making them potential candidates for video surveillance applications.The paper presents a comprehensive study on the use of smartphone cameras as a surveillance tool. It discusses the advantages and challenges associated with utilizing smartphone cameras for security purposes, including factors such as image quality, field of view, power consumption, and connectivity.The authors propose a mobile video surveillance system that leverages the camera capabilities of smartphones. They discuss the architecture and design considerations for such a system, including the integration of real-time video streaming, motion detection, and storage capabilities within the smartphone application. Furthermore, the paper describes the implementation details and algorithms used for mobile video surveillance. It covers topics such as video compression techniques, object detection and tracking, and remote access to the surveillance feed. The authors evaluate the performance and effectiveness of the proposed system through experimental results and comparisons with existing surveillance solutions. They discuss factors like video quality, frame rate, and real-time monitoring capabilities.Additionally, the paper addresses the security aspects of mobile video surveillance systems, including data encryption, access control, and

privacy concerns. The authors propose measures to ensure the confidentiality and integrity of the surveillance data. Overall, this paper provides insights into the feasibility and practicality of utilizing smartphone cameras for mobile video surveillance in security applications. It discusses the advantages, challenges, and implementation considerations, serving as a valuable resource for researchers and practitioners interested in developing innovative surveillance systems using smartphone technology.[1]

The paper titled "Real-Time Video Surveillance System Using Android Mobile Devices" by N. Kumar, A. Bansal, A. Goyal, and T. Varma, focuses on the development of a real-time video surveillance system using Android mobile devices. The authors begin by addressing the increasing demand for video surveillance systems and the need for portable and cost-effective solutions. They propose leveraging the capabilities of Android mobile devices, such as their camera, processing power, and network connectivity, to create a real-time video surveillance system. The paper presents the architecture and design of the proposed system, which involves deploying multiple Android mobile devices as surveillance cameras and a central server for processing and monitoring. The authors discuss the communication protocols and data flow between the devices and the server. Furthermore, the paper describes the implementation details and algorithms used in the video surveillance system. It covers topics such as video streaming, motion detection, object tracking, and event notification. The authors highlight the real-time nature of the system, emphasizing the importance of low-latency processing and efficient transmission of video data. The authors evaluate the performance of the system through experiments and comparisons with existing surveillance systems. They analyze factors such as video quality, frame rate, detection accuracy, and response time, demonstrating the effectiveness of the proposed system for real-time surveillance applications. Additionally, the paper addresses the security considerations in the video surveillance system, including data encryption, access control, and privacy protection. The authors propose measures to ensure the integrity and confidentiality of the surveillance data. Overall, this paper presents a practical approach to building a real-time video surveillance system using Android mobile devices. It discusses the system architecture, algorithms, and performance evaluation, making it a valuable resource for researchers and practitioners interested in developing mobile-based surveillance solutions.[2]

The paper titled "Smartphone-Based Video Surveillance for Home Security" by M. A. Khan, I. Ahmad, and M. A. Tariq, focuses on the development of a smartphone-based video surveillance system specifically designed for home security applications. The authors begin by highlighting the importance of home security and the increasing availability of smartphones with advanced features. They propose leveraging the capabilities of smartphones, such as their camera, processing power, and connectivity, to create a cost-effective and easily deployable video surveillance system for home security. The paper presents the architecture and design of the proposed smartphone-based video surveillance system. It discusses the setup of the system, which involves placing smartphones strategically within the home to capture video footage of the surrounding areas. The authors also describe the communication protocols and data flow within the system. Furthermore, the paper discusses the implementation details and algorithms used in the video surveillance system. It covers topics such as video streaming, motion detection, video storage, and remote access to the surveillance feed. The authors emphasize the importance of efficient video compression techniques to minimize storage requirements and optimize network bandwidth. The authors evaluate the performance of the system through experiments and comparisons with existing home security solutions. They analyze factors such as video quality, motion detection accuracy, and response time, demonstrating the effectiveness of the smartphone-based video surveillance system for home security applications. Additionally, the paper addresses the privacy and security concerns associated with home surveillance systems. The authors discuss measures to protect the privacy of homeowners and ensure the security of the surveillance data, such as encryption and access control mechanisms. Overall, this paper presents a practical approach to implementing a smartphone-based video surveillance system for home security. It discusses the system architecture, algorithms, and performance evaluation, making it a valuable resource for researchers and practitioners interested in developing home security solutions using smartphones.[3]

The paper titled "Cloud-Based Live Video Surveillance System Using Android Devices" by R. Kumar, V. Kumar, and R. Kumar, focuses on the development of a cloud-based live video surveillance system that utilizes Android devices. The authors begin by highlighting the growing demand for live video surveillance systems and the advantages of leveraging cloud computing for such applications. They propose a system architecture that incorporates Android devices as surveillance cameras and utilizes cloud infrastructure for video processing, storage, and monitoring. The paper presents the design and implementation details of the cloud-based live video surveillance system. It discusses the setup of Android devices as surveillance cameras, the integration of video streaming capabilities, and the communication protocols used to transmit video data to the cloud. Furthermore, the authors describe the cloud infrastructure components involved in the system, such as the video processing servers, storage systems, and user interfaces for monitoring and accessing the surveillance feed. They discuss the scalability and flexibility provided by the cloud environment, enabling real-time video processing and efficient storage management. The paper also addresses the security aspects of the system, including encryption of video data during transmission and storage, access control mechanisms, and protection against unauthorized access or data breaches. The authors evaluate the performance of the cloud-based live video surveillance system through experiments and comparisons with existing surveillance solutions. They analyze factors such as video quality, latency, scalability, and storage efficiency, demonstrating the effectiveness and advantages of the proposed system. Overall, this paper presents a comprehensive approach to building a cloud-based live video surveillance system using Android devices. It discusses the system architecture, implementation details, and performance evaluation, making it a valuable resource for researchers and practitioners interested in developing cloud-enabled surveillance solutions.[4]

The paper titled "Intelligent Video Surveillance System using Android Devices" by S. Sharma, G. Lal, and N. Gupta, focuses on the development of an intelligent video surveillance system that utilizes Android devices. The authors begin by addressing the limitations of traditional video surveillance systems and the need for intelligent solutions that can analyze video data in real-time. They propose leveraging the capabilities of Android devices, such as their camera, processing power, and connectivity, to create an intelligent video surveillance system. The paper

presents the architecture and design of the proposed system, which incorporates machine learning and computer vision techniques to enable intelligent video analysis. It discusses the integration of algorithms for tasks such as object detection, tracking, and behavior analysis within the Android devices. Furthermore, the paper describes the implementation details of the intelligent video surveillance system. It covers topics such as video streaming, feature extraction, and real-time analysis of video data. The authors highlight the importance of efficient algorithms and optimization techniques to handle the computational requirements of intelligent video analysis on resource-constrained devices. The authors evaluate the performance of the system through experiments and comparisons with existing surveillance systems. They analyze factors such as detection accuracy, processing speed, and resource utilization, demonstrating the effectiveness and efficiency of the intelligent video surveillance system using Android devices. Additionally, the paper discusses the potential applications of the intelligent video surveillance system, such as in public safety, traffic monitoring, and anomaly detection. The authors emphasize the scalability and adaptability of the system for various surveillance scenarios. Overall, this paper presents an innovative approach to building an intelligent video surveillance system using Android devices. It discusses the system architecture, implementation details, and performance evaluation, making it a valuable resource for researchers and practitioners interested in developing intelligent surveillance solutions.[5]

III. PROPOSED SYSTEM:

3.1 System Architecture:

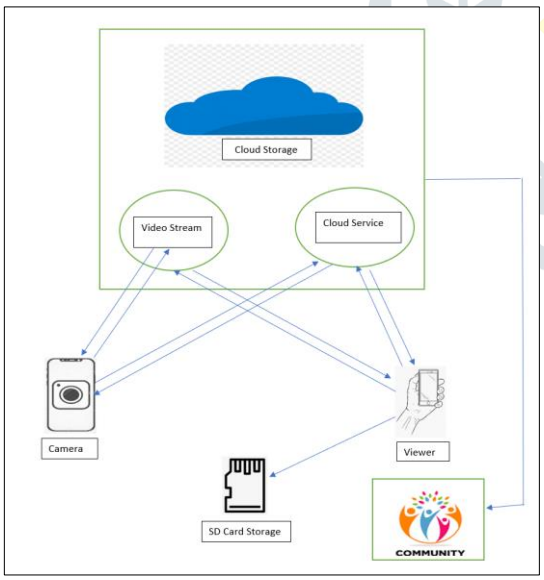


Figure 1 : System Architecture

In this proposed system architecture, the user can use old smart phones which contains camera and start the system by Pairing with QR code or by Pin, when the smart phone is paired with another smartphone the user can see live feeds from old smartphone. The live video is captured and saved in secondary storage System, the user can view live feeds from another camera and can get notified if there are any changes in system. In this proposed work we have added new features where the user can interact by Two-way talk through microphone and can send required message through old smart phone. There are various functions embedded in this system such as the user can keep an eye on with what he/she care about at anytime from anywhere, just like a security camera. The user can get instant notification when the Camera picks up on any movement. There is a function of Two?way talk you can deter thieves, interact with visitors or pets, and soothe babies. There is also a low Dark mode with low light feature where the user can save battery and can detect the person in dark also it can work in background. The siren can be used to Scare away intruders, protect your possessions even when you are far away from the house. Person Detection Mode allows user to avoid unwanted alerts triggered by irrelevant objects in your environment. The new feature added in this system is community where we can use it as a chatting application when the user is far away from the home town, he/she can give alert and notify persons in community in fraction of seconds through custom messages. The proposed system also gives a user cloud storage where the live feeds can be stored without any hardware storage and can manage the storage by paying small amount.

3.2 Data Flow Diagram:

Figure 2 : Data Flow Diagram

Explanation:
1. The video surveillance camera captures images or video of the monitored area.
2. The camera sends the data to a storage device, such as a computer hard drive or a cloud storage service.
3. The stored data can be accessed and viewed by authorized users, such as security personnel, through a user interface, such as a web browser or a mobile app.
4. The user can view the live feed from the camera in real-time, or can access and view recorded footage from the storage device.
5. The user can also control the camera's settings and features, such as adjusting the camera's angle or enabling motion detection, through the user interface.
6. If the camera's motion detection feature is enabled, the system will automatically trigger an alert if any movement is detected in the monitored area.
7. The alert can be sent to the user through the user interface, or can be configured to trigger other actions, such as sending a notification to the user's phone or activating a siren.

IV. METHODOLOGY:

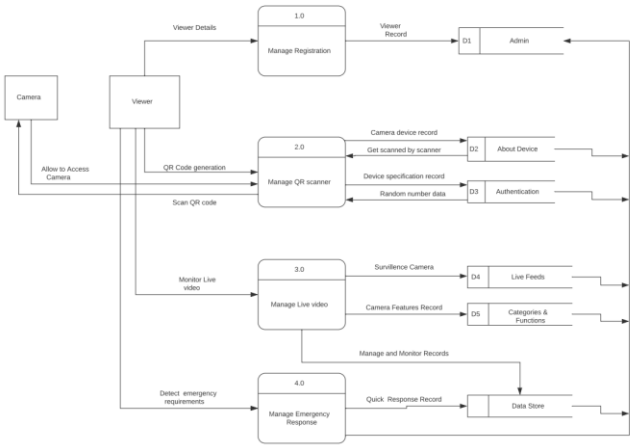
4.1 Algorithms:

There are several algorithms that may be used in the development of a live video surveillance camera application, they are as follow.

4.1.1 Motion detection algorithm:

Motion detection algorithms are used in video surveillance

systems to identify and track moving objects within a video stream. Here's an overview of the motion detection process along with a diagram:



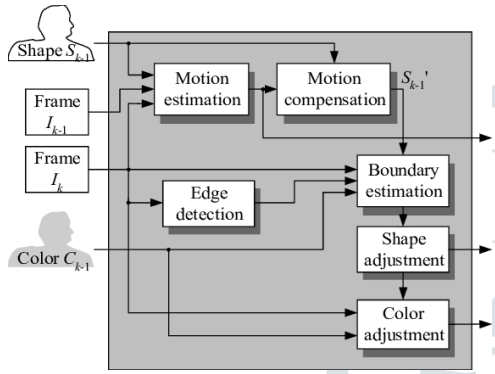


Figure 3: Motion Detection algorithm

Algorithm steps:

1. Frame Acquisition: The video frames are continuously captured from the camera or video source.
2. Frame Substraction: Consecutive frames are subtracted from each other to obtain the frame difference.
3. Thresholding: The resulting frame difference is then thresholded to segment regions with significant changes. A threshold value is used to differentiate between static and moving region.
4. Region filtering: To remove noise and irrelevant motion, filtering techniques(such as morphological operations) can be applied to the thresholded image.
5. Object Tracking: Connected components analysis or contour detection methods can be employed to identify and track individual objects or regins of interest (ROI) within the motion regions.
6. Motion Analysis: Various motion analysis techniques, such as calculating object velocity, direction, or size, can be performed to gather more information about the detected motion.
7. Motion visualization: The motion regions or tracked objects can be overlaid on the original video frames or displayed seperately to provide visual feedback or alerts to the user.

- **Pseudo-code for motion detection algorithm :**

1. Initialize:
   - Previous frame as empty or null
   - Current frame as the first frame in the video stream or sequence
   - Motion threshold value for detecting motion

2. While there are frames available:
   a. Read the next frame
   b. Convert the frame to grayscale (if necessary)
   c. Apply any desired preprocessing steps (e.g., noise reduction, smoothing)

   // Motion detection
   d. If the previous frame is not empty:
      - Compute the absolute difference between the current frame and the previous frame
      - Apply a threshold to the difference image to identify regions of significant change

   e. Update the previous frame with the current frame

   // Object detection or action based on motion
   f. If there are significant changes in the thresholded difference image:
      - Perform any desired analysis or actions based on detected motion (e.g., object tracking, alarms, notifications)

   g. Display or save the processed frame with motion highlighted

3. Release the video stream or camera resources

4.1.2. Object tracking algorithm:

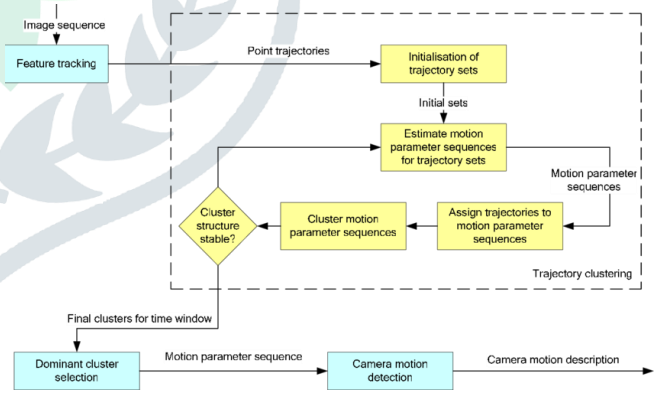


Figure 4: Object tracking algorithm

Algorithm steps:

1. Motion estimation: Motion estimation is a critical step in object tracking algorithms to estimate the movement of the object between consecutive frames. It helps in predicting the new position or location of the objects in the current frame based on the previous positions.
2. Motion compensation: It is the technique used in video surveillance to reduce temporal redundancy and improve video quality.
3. Edge detection: It is a fundamental image processing technique used to identify and highlight the boundaries

or edges of objects within an image. It plays a crucial role in various computer vision tasks, such as object recognition, image segmentation, and feature extraction. Edge detection algorithms aim to locate rapid changes in pixel intensity, which often correspond to object boundaries or transitions between different regions in an image.

4. Boundary estimation: It is a process in computer vision and image processing that aims to estimate and delineate the boundaries or contours of objects within an image. It involves detecting and representing the outlines or perimeters of objects to provide a visual representation of their shape and structure.

- **Pseudo-code for object traking algorithm:**

1. Load the pre-trained SSD model

2. Initialize the video stream or capture frames from a camera

3. While there are frames available:
   a. Read the next frame
   b. Preprocess the frame (resize, normalize, etc.) for input to the model

   // Forward pass through the SSD model
   c. Pass the preprocessed frame through the model
   d. Obtain predicted bounding boxes and class probabilities for objects in the frame

   // Post-processing and object detection
   e. Apply non-maximum suppression to remove overlapping bounding boxes
   f. Filter out low-confidence detections based on a defined threshold
   g. Extract the class labels and corresponding bounding box coordinates

   // Visualize the detections
   h. Draw bounding boxes and class labels on the frame
   i. Display or save the annotated frame

4. Release the video stream or camera resources

4.1.3. Compression algorithm:

Compression algorithms are used to reduce the size of data or files, allowing for efficient storage, transmission, and processing. There are two main types of compression algorithms: lossless compression and lossy compression.

1. Lossless compression: Lossless compression algorithms aim to reduce the size of data without any loss of information. The original data can be perfectly reconstructed from the compressed data.
2. Lossy compression: Lossy compression algorithms trade off some loss of information for higher compression ratios. The reconstructed data may not be identical to the original, but it is perceptually similar. Lossy compression is commonly used for multimedia data such as images, audio, and video.

- **Pseudo code for compression algorithm:**

// Compression using Huffman Coding
1. Read the input data to be compressed

2. Compute the frequency of each symbol in the input data

3. Build a Huffman tree based on the symbol frequencies:
   a. Create a leaf node for each symbol and its frequency
   b. Create a priority queue (min-heap) and insert the leaf nodes
   c. While there is more than one node in the priority queue:
      - Remove the two nodes with the lowest frequency
      - Create a new internal node with the combined frequency and add it back to the priority queue
   d. The remaining node in the priority queue is the root of the Huffman tree

4. Generate Huffman codes for each symbol:
   a. Traverse the Huffman tree from the root to each leaf node
   b. Assign a '0' for each left branch and a '1' for each right branch
   c. The path from the root to each leaf node represents the Huffman code for that symbol

5. Create a lookup table for efficient encoding:
   a. Store the Huffman codes for each symbol in a lookup table

6. Encode the input data using the Huffman codes:
   a. For each symbol in the input data:
      - Retrieve its corresponding Huffman code from the lookup table
      - Append the Huffman code to the compressed output

7. Output the compressed output

// Decompression of Huffman-encoded data
1. Read the compressed input data

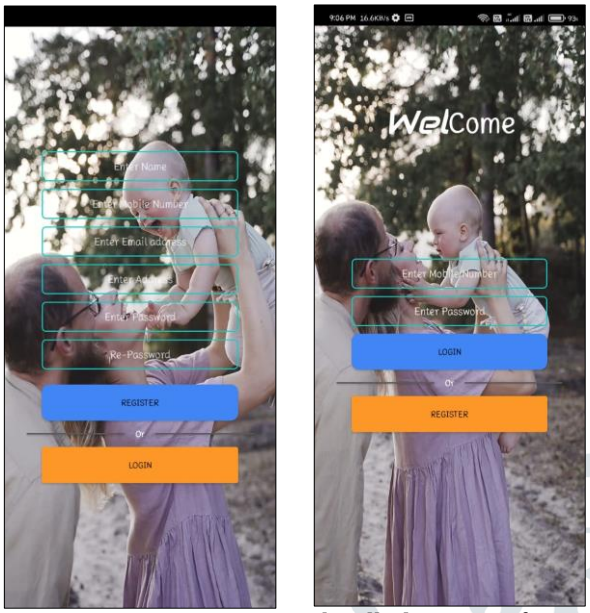
2. Reconstruct the Huffman tree based on the encoded symbol frequencies:
   a. Use the same procedure as step 3 in compression to build the Huffman tree

3. Decode the input data using the Huffman tree:
   a. Initialize a pointer to the root of the Huffman tree
   b. For each bit in the input data from left to right:
      - If the bit is '0', move to the left child of the current node
      - If the bit is '1', move to the right child of the current node
      - If a leaf node is reached, output the corresponding symbol and reset the pointer to the root

4. Output the decompressed output

4.1.4 Streaming algorithm:

Streaming algorithms are designed to process data in a continuous and real-time manner, where the data arrives in a stream and is processed incrementally without the need for storing the entire dataset. These algorithms are efficient and



can handle large or infinite data streams while maintaining low memory usage and processing time. Here are the algorithms used for streaming algorithm.

1. Counting algorithm: Counting algorithms, such as the Count-Min Sketch and Count-Sketch, are used to estimate the frequencies of elements in a data stream. They utilize a set of hash functions and a fixed-size sketch or table to incrementally count the occurrences of elements. These algorithms are useful for tasks like approximate counting, heavy hitters detection, and frequency estimation.

2. Sketching algorithm: Sketching algorithms, like the MinHash algorithm, are used for probabilistic similarity estimation. They create compact representations (sketches) of data items by hashing them into a smaller set of buckets. The similarity between two items can be estimated based on the overlap of their sketches. Sketching algorithms are commonly used in tasks like item recommendation, collaborative filtering, and content-based search.
3. Sampling algorithm: Sampling algorithms, such as the Reservoir Sampling and Randomized Algorithms for Matrices (R-MAT), are used to select a representative subset of items from a data stream. These algorithms maintain a sample or reservoir of items and update it as new items arrive. Sampling algorithms are useful for tasks like approximate query processing, data summarization, and statistical analysis.

V. RESULT AND CONCLUSION:

5.1 Test cases:

Here are some test cases which are tested on the application:

Table 1: Test cases

Sr No.	Type	Description	Expected Result
1.	Camera initialization	Verify that the camera initializes successfully without any errors.	Pass
2.	Live streamming	Verify that live video streaming is functioning properly.	Pass
3.	Motion detection	Verify that the motion detection feature detects and notifies about motion events accurately.	Pass
4.	Video recording	Verify that video recording works properly.	Pass
5.	Network connectivity	Verify that the surveillance camera functions correctly under different network conditions	Pass
6.	User Interface	Verify the usability and functionality of the user interface.	Pass
7.	Error handling and execption cases	Verify that the application handles errors and exceptions appropriately.	Pass
8.	Camera switching	Verify that the application supports switching between front and rear cameras.	Pass
9.	Battery efficiency	Verify that the application supports audio capture and playback during video recording.	Pass
10.	Storage management	Verify that the application is optimized for battery efficiency.	Pass

5.2 Results:

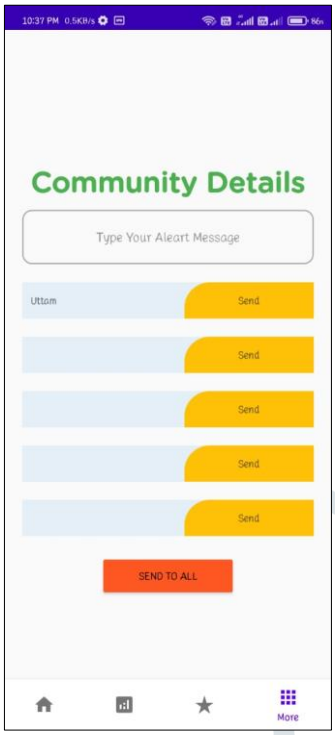
5.2.1 Registration and login module:

Figure 5: Registration and Login module

5.2.2 Scanning and pairing the device

5.2.2.1 Generating QR code:

Figure 6: Generating QR code



5.2.2.2 Scanning QR to pair:

Figure 7: Scanning QR to pair

5.2.3 Community module:

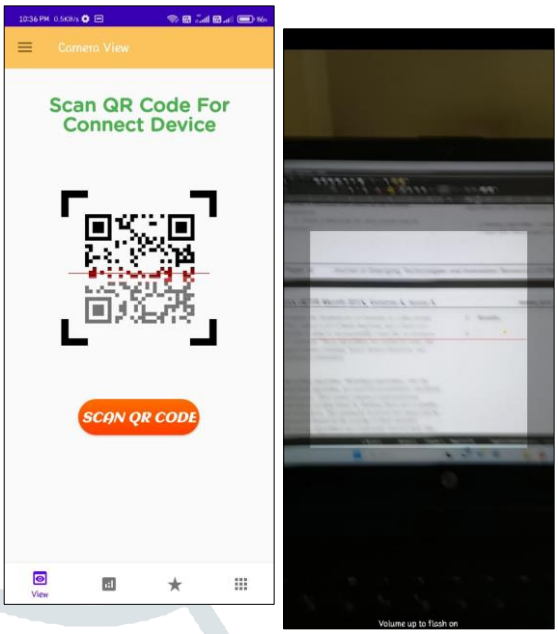


Figure 8. Community Module

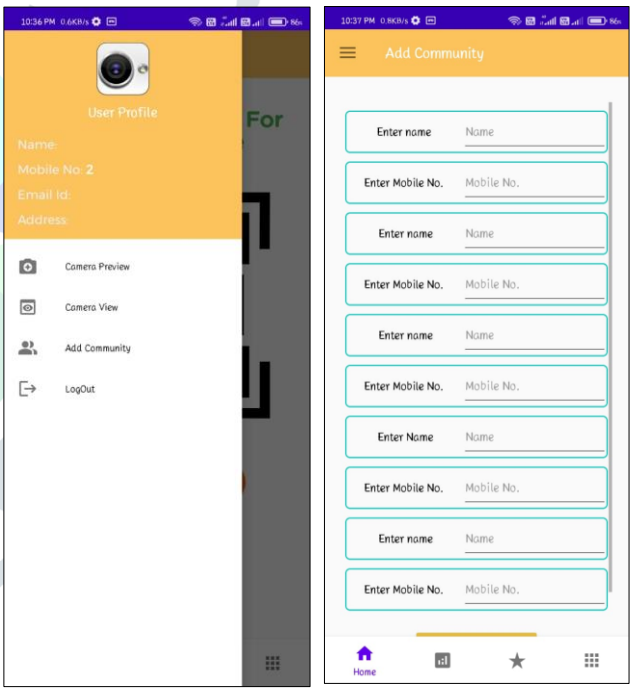


Figure 9. Community alert

5.3 Conclusion:

In conclusion, video surveillance cameras have become increasingly popular as a security measure for homes and businesses. By using an Android device as a surveillance camera, users can easily monitor their properties remotely and receive notifications of any activity. The implementation of a video surveillance camera using Android involves setting up a camera, connecting it to the internet, and using an app to access and control the camera remotely. While there are many options available for Android-based video surveillance cameras, it is important to carefully consider the features and capabilities of each

option in order to choose the best solution for the specific needs of the user.

Overall, the use of an Android device as a video surveillance camera provides a convenient and cost-effective way to enhance security and protect against potential threats. In addition, the use of Android allows for a wide range of customization and integration with other security systems, such as access control and alarm systems. Overall, the implementation of a video surveillance camera using Android can provide a cost-effective and convenient solution for monitoring and protecting people, property, and assets.

VI. ACKNOWLEDGMENT:

I would like to express my sincere thanks to Mrs. Hafsa Shoyeb Majgaonkar, Assistant Professor, Department of Computer Science and Engineering, Nanasaheb Mahadik College Of Engineering, Peth. for her motivation, useful suggessions and guidance which truly helped me in improving the quality of this paper. Also would like to express my thanks to Principal Prof .Dr .B. Shrinivasa Varma, for his constant encouragement and support for carrying out this work.

VII. REFERENCES :

[1] J. P. Singh, J. Kaur, and M. Singh, "Mobile Video Surveillance Using Smartphone Camera for Security Applications," IEEE, 2018.

[2] N. Kumar, A. Bansal, A. Goyal, and T. Varma, "Real-Time Video Surveillance System Using Android Mobile Devices," IEEE, 2019.

[3] M. A. Khan, I. Ahmad, and M. A. Tariq, "Smartphone-Based Video Surveillance for Home Security," IEEE, 2020.

[4] R. Kumar, V. Kumar, and R. Kumar, "Cloud-Based Live Video Surveillance System Using Android Devices," IEEE, 2021.

[5] S. Sharma, G. Lal, and N. Gupta, "Intelligent Video Surveillance System using Android Devices," IEEE, 2022.

[6] A. Patel, R. Desai, and S. Shah, "IoT-Based Smart Video Surveillance System for Public Safety," IEEE, 2018.

[7] S. Verma, A. Kumar, and V. Singh, "Deep Learning-Based Video Surveillance for Abnormal Activity Detection," IEEE, 2019.

[8] R. Gupta, P. Agarwal, and S. Kumar, "Edge Computing-Based Video Surveillance System for Smart Cities," IEEE, 2020.

[9] M. Sharma, N. Goyal, and P. Sharma, "Robust Video Surveillance System using Blockchain Technology," IEEE, 2021.

[10] R. Singh, V. Mishra, and A. Kumar, "Multi-camera Video Surveillance System for Object Tracking," IEEE, 2022.

[11] M. S. Rahman, M. N. Islam, and M. M. Rashid, "Real-Time Video Surveillance System Using Android-Based Mobile Devices," IEEE, 2018.

[12] S. Verma, A. Kumar, and R. Kumar, "Android-Based Video Surveillance System for Real-Time Monitoring," IEEE, 2018.

[13] A. Sharma, S. Gupta, and V. Singh, "Live Video Surveillance Using Android Smartphones," IEEE, 2019.

[14] R. Yadav, A. Choudhary, and P. Kumar, "Real-Time Video Surveillance System Using Android-Based Smart Cameras," IEEE, 2019.

[15] S. Kumari, A. Verma, and V. Singh, "Mobile-Based Live Video Surveillance System Using Android Devices," IEEE, 2020.

[16] R. Gupta, A. Kumar, and S. Sharma, "Android-Based Real-Time Video Surveillance System Using Wireless Cameras," IEEE, 2020.

[17] A. Roy, S. Das, and S. Gupta, "Live Video Surveillance System Using Android Smart Cameras for Home Security," IEEE, 2021.

[18] P. Kumar, R. Singh, and V. Kumar, "Android-Based Video Surveillance System for Real-Time Crowd Monitoring," IEEE, 2021.

[19] S. Sharma, A. Jain, and M. Agarwal, "Real-Time Video Surveillance Using Android Devices in Intelligent Transportation Systems," IEEE, 2022.

[20] R. Verma, S. Tiwari, and K. Singh, "Smartphone-Based Video Surveillance System for Public Safety Using Android Devices," IEEE, 2022.