**Literature Survey**

**ECE SEC – 2**

**BATCH - 1**

**PROJECT TITLE**

# **REAL TIME IMAGE STREAMING FROM ESP32 CAM MODULE TO TELEGRAM CHATBOT**

STUDENT ID STUDENT NAME SIGNATURE

2310040095 - SAYOOJ S -----------------

2310040076 - VENKAT PRASAD ------------------

2310040089 - SAITEJA ------------------

2310040132 - RAVI RATNA ------------------

### ****Methodology: Real-Time Image Streaming from ESP32-CAM to Telegram Chatbot****

#### ****1. Introduction****

This section outlines the step-by-step methodology for implementing a real-time image streaming system using the ESP32-CAM module, which captures images and transmits them via a Telegram chatbot for remote monitoring.

#### ****2. System Architecture****

The project follows a modular approach, consisting of the following components:

* **ESP32-CAM Module**: Captures real-time images.
* **Wi-Fi Communication**: Sends images over the internet.
* **Telegram Bot API**: Facilitates remote communication.
* **Microcontroller Programming**: Implements face detection and user authentication.
* **Solenoid Lock (Optional)**: Controls access based on image verification.

#### ****3. Hardware Implementation****

##### ****3.1 Components Used****

* **ESP32-CAM Module**
* **Power Supply (5V)**
* **Solenoid Lock (Optional)**
* **MicroSD Card (For Local Storage, Optional)**
* **Relay Module (For Lock Control, Optional)**

##### ****3.2 Circuit Design****

* The ESP32-CAM is connected to a power source.
* The camera module captures images when motion or a face is detected.
* The solenoid lock is connected via a relay module (if used).

#### ****4. Software Implementation****

##### ****4.1 Programming Environment****

* **Arduino IDE / PlatformIO**: Used for writing and uploading firmware.
* **Telegram Bot API**: Used for communication between ESP32-CAM and the user.
* **Python / Node.js (Optional)**: For advanced image processing.

##### ****4.2 Steps for Software Development****

1. **ESP32-CAM Setup**
   * Configure the ESP32-CAM module with the necessary firmware.
   * Connect to Wi-Fi using SSID and password.
2. **Image Capture and Processing**
   * Use the ESP32-CAM's camera library to capture images.
   * Store images temporarily in the microcontroller's memory.
3. **Telegram Bot Integration**
   * Create a Telegram bot using BotFather.
   * Obtain an API key and integrate it with the ESP32-CAM firmware.
   * Use HTTP requests to send images to Telegram.
4. **Face Detection (Optional)**
   * Implement Haar cascade or deep learning-based detection.
   * Authenticate users before triggering the door lock.
5. **Remote Access and Control**
   * Send notifications to the user when an image is captured.
   * Allow users to manually unlock the door via Telegram commands.

#### ****5. Testing and Validation****

* **Unit Testing**: Verify each component (camera, Wi-Fi, Telegram integration).
* **System Testing**: Ensure real-time image transmission without delay.
* **Security Testing**: Check for unauthorized access attempts.

#### ****6. Challenges and Future Scope****

* **Challenges**:
  + Wi-Fi connectivity issues.
  + Image transmission delays.
  + Power consumption optimization.
* **Future Enhancements**:
  + AI-based face recognition for automated authentication.
  + Cloud storage for image logging.
  + Integration with other smart home devices.