Software Design Specification

Skill Development Project III – ICT 3206

Bachelor of Information and Communication Technology (Honors)

Department of Information and Communication Technology
Faculty of Technology
Rajarata University of Sri Lanka

Details of the Project

Project Title : Intelligent Indoor OR Outdoor Surveillance Camera with AI Detection and Programmable Relay Control

Group Number : Group 01

Group Name : Tech Titans

Submission Date : 2024/09/15

Details of the Group Members

Name	Registration ID	Index No.	Signature
E.M.S.M. Edirisooriya	ITT/2020/021	1297	
D.H.D. Prabhasha	ITT/2020/078	1348	Dumindu.
H.L.I.N. Liyanaarachchi	ITT/2020/128	1390	
P.W.D.I.M. Rodrigo	ITT/2020/091	1357	
B.M.K.L. Bandaranayake	ITT/2020/121	1383	Niranga
K.R.N. Perera	ITT/2020/072	1343	Romes
A. Sanjayan	ITT/2020/097	1363	

Name : Ms. P.R.H.N.G. Thilakarathne

Designation : Lecturer

Department : Information & Communication Technology

Contact Details : +94 71 954 5073

Table of Contents

List of Figures

1 Introduction

1.1 Background of the project

Home security is becoming increasingly important as technology advances. Traditional CCTV surveillance systems face several challenges, such as high costs, complex installation, limited functionality, and accessibility issues. These older security camera systems often lack the flexibility and advanced features needed to handle today's security challenges effectively.

1.2 Purpose and significance of the project

This project aims to address above mention issues by developing an intelligent indoor/outdoor surveillance camera system integrated with AI detection and programmable relay control. The system utilizes artificial intelligence, cloud technology, and a mobile app to offer a comprehensive security solution. By combining smart object detection, live video streaming, video recording, and relay control, this system provides excellent monitoring and control capabilities. The development of a mobile app using Flutter ensures that users can easily interact with and manage the system from their phones.

The significance of this project lies in its ability to provide a cost-effective, user-friendly, and scalable solution for enhanced security and monitoring. It aims to overcome the limitations of traditional CCTV systems by leveraging IoT technology, thereby delivering a reliable and user- friendly surveillance solution that meets today’s security needs.

1.3 Scope of the project

The scope of this project encompasses the design, development, and implementation of an AI-powered ESP32-CAM WIFI IP camera surveillance system with advanced functionalities for indoor and outdoor monitoring. The primary components and functionalities to be developed include:

- **ESP32-CAM WIFI IP Camera**: Design and setup of the ESP32-CAM module, integrating a 2MP camera for video recording.
- **AI-Powered Object Detection**: Implementation of the YOLOv8 object detection algorithm to identify and classify objects in real-time, with optimization for efficient and accurate detection.
- **Cloud-Based Live Streaming**: Development of a secure cloud-based solution for streaming live video feeds.
- **Video Recording and Playback**: Functionality to start and stop video recordings from the mobile application, storage on the ESP32-CAM's SD card or the user's mobile device, and playback features within the mobile application.
- **Programmable Physical Relay Control**: Design and implementation of a dual relay module to control external devices (e.g., lights, alarms, locks), integrated within the mobile application for remote management.

- **Flutter Mobile Application**: Development of a cross-platform mobile application using Flutter, with user-friendly interfaces for monitoring live video feeds, receiving alerts, and controlling physical relays.
- **Security Alerts and Notifications**: Intelligent mechanisms for analyzing events and providing context-aware alerts, customizable by users.

1.4 Objectives of the Project

- Develop the ESP32-CAM WIFI IP Camerato integrate seamlessly with live streaming, recording, and relay control functionalities of the surveillance system.
- Integrate AI-Powered Object Detection using YOLOv8 for accurate identification and classification of objects in real-time.
- Enable Cloud-Based Live Streaming for remote access to the surveillance camera's live video feed with robust security measures.
- Implement Video Recording Functionality allowing users to start and stop recordings from the mobile app.
- Implement Remote Physical Relay Control enabling users to manage external devices (e.g., lights, alarms) remotely via the mobile app.
- Develop a Flutter-based Mobile Application with user-friendly interfaces for monitoring live video, receiving security alerts, and controlling connected devices.
- Enhance Security Alerts and Notifications with intelligent mechanisms for analyzing events and providing context-aware alerts customizable by users.

1.5 System design approach

1. Software Development Process Model

For our "Intelligent Indoor/Outdoor Surveillance Camera with AI Detection and Programmable Relay Control" project, we use **SCRUM** models due to the iterative and flexible nature of development:

- **SCRUM**: SCRUM is an Agile framework with defined roles such as Product Owner, SCRUM Master, and Development Team. It uses sprints (2-4 weeks) to deliver incremental features. Regular meetings (stand-ups, sprint reviews) keep the team aligned and allow progress monitoring, as we are having meetings with our team and mentor.

2. Design Patterns

- **Model-View-Controller (MVC)**: This pattern is suitable for our mobile app. The **Model** represents the logic (e.g., AI detection and relay control), the **View** handles the UI (e.g., monitoring video, controlling devices), and the **Controller** manages the flow between them. This separation of concerns improves maintainability.
- **Client-Server Architecture**: Since our system involves live streaming and cloud-based access, a client-server model is apt. The **ESP32-CAM** (camera) serves as the server, streaming video to the mobile app (client) over the internet, with remote control features managed via the server. This design pattern supports scalable, secure remote surveillance.

2 Architectural Design

2.1 System Architecture

The system architecture for the surveillance system is based on an integration of hardware and software components that function together to provide an intelligent, real-time security solution.

Overview:

The system includes the following key components

- **ESP32-CAM Module**: A low-power microcontroller with an integrated 2MP camera for video capture.
- **YOLOv8 Object Detection Model**: Embedded AI model used to identify and classify objects in real time.
- **Cloud Server**: Handles storage, cloud-based live streaming, and access control.
- **Mobile Application (Flutter)**: Allows users to interact with the system, view live feeds, control devices, and receive alerts.
- **Relay Module**: Controls external devices such as lights or alarms based on input from the AI system or user.

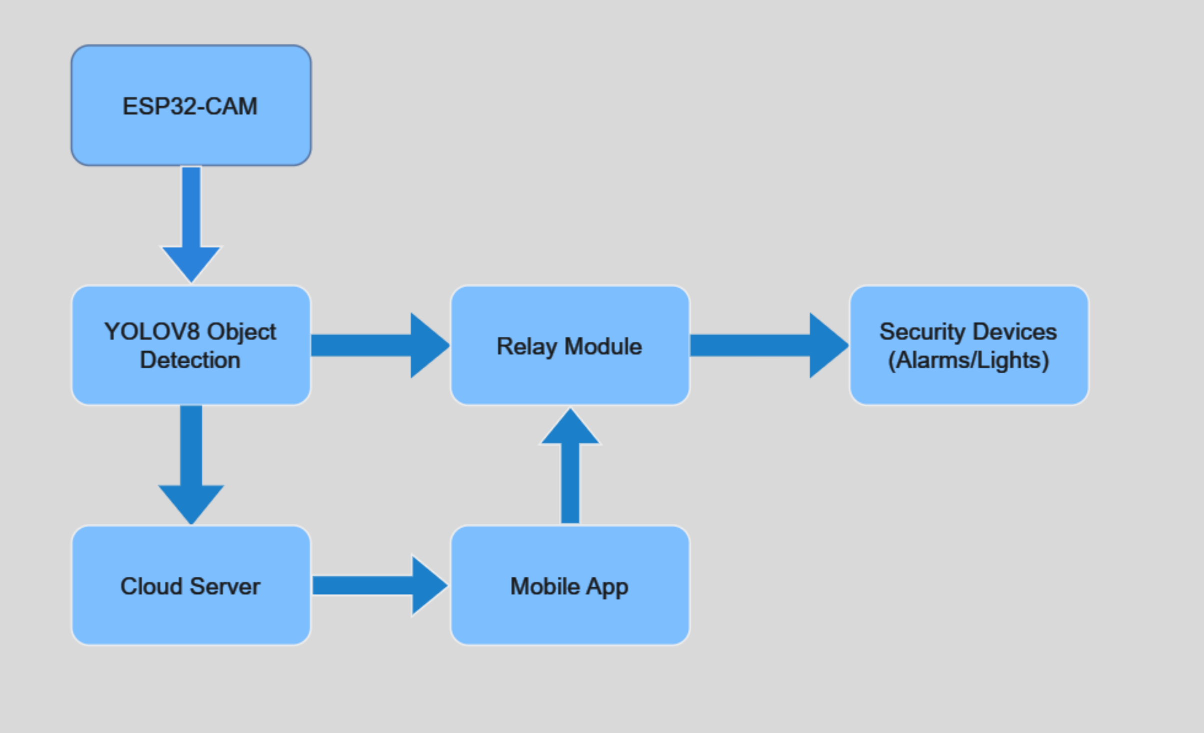


Figure 1 block diagrams

2.2 Component design

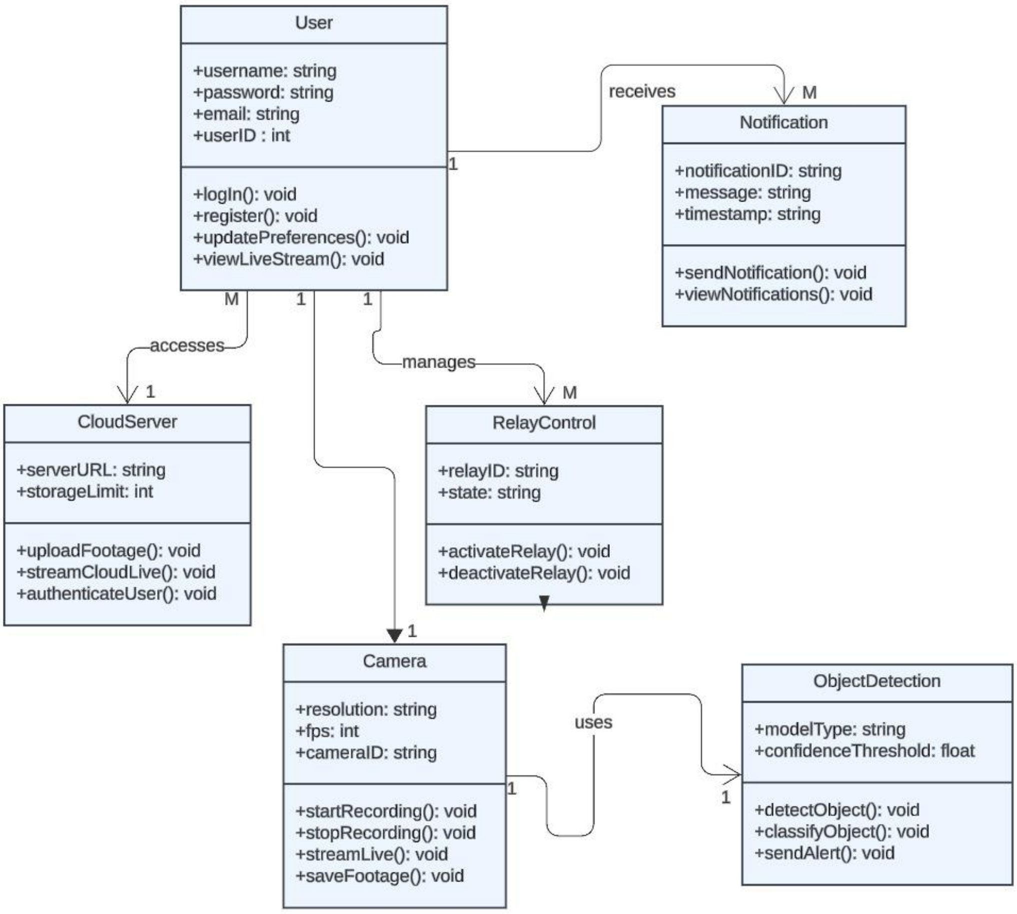


Figure 2 class diagram

Description:

- The **Camera Class** handles video feed management, enabling live streaming or saving footage based on user commands.
- The **Object Detection Class** integrates the YOLOv8 AI model, running detection algorithms on captured video frames and sending alerts if an object is recognized.
- The **User Class** manages user authentication and settings, allowing users to access live streams, recorded footage, and security preferences.
- The **Relay Control Class** facilitates the management of external security devices such as lights or alarms, allowing remote activation or deactivation.
- The **Cloud Server Class** is responsible for storing video recordings and enabling cloud-based live streaming, along with secure user access.
- The **Notification Class** handles alert generation, enabling the system to send push notifications to the user's mobile app when a detection event occurs.

2.3 Processes and interaction design

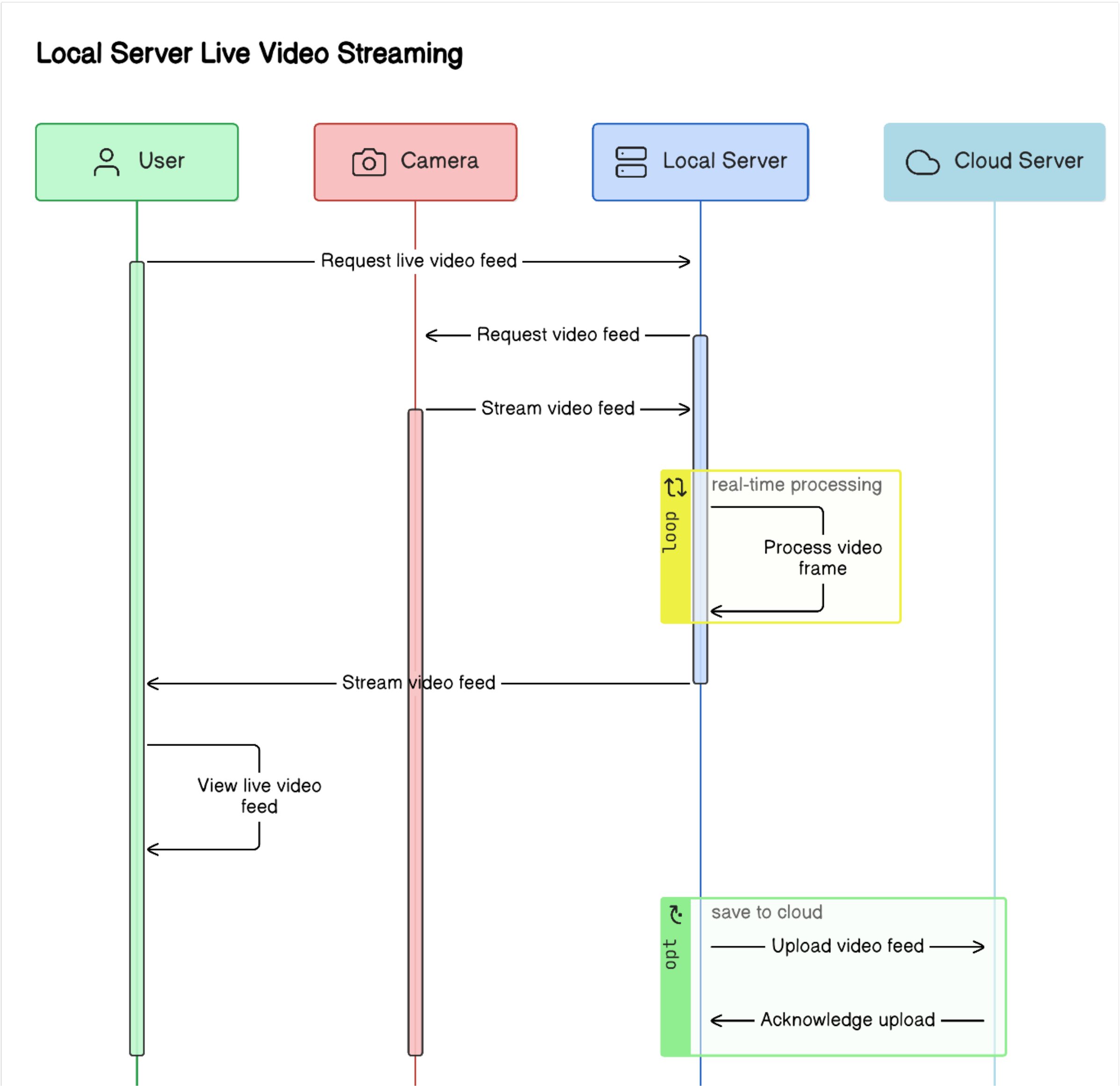


Figure 3 sequence diagrams for Local Server Live Video Streaming

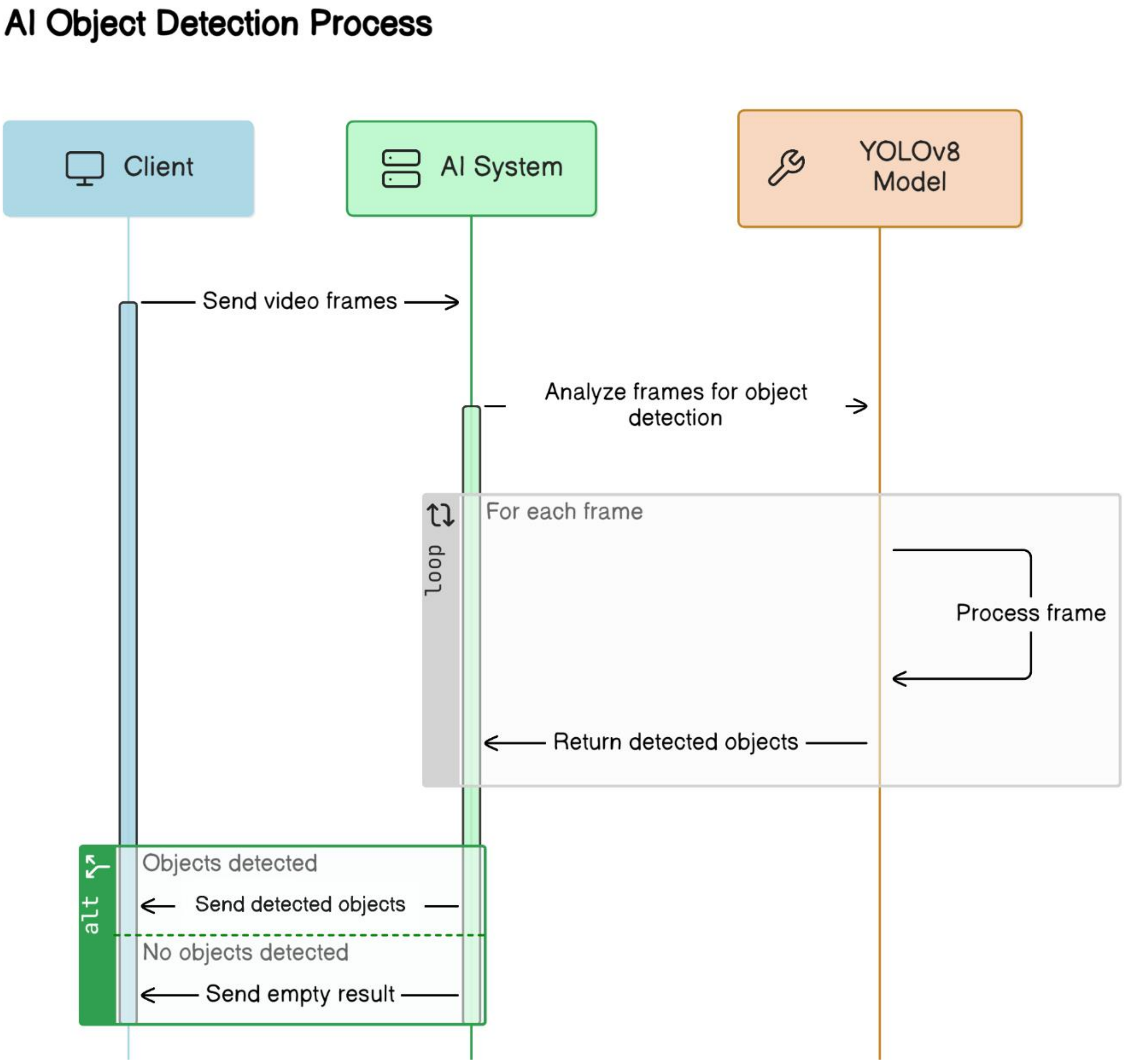


Figure 4 sequence diagrams for AI Object Detection Process

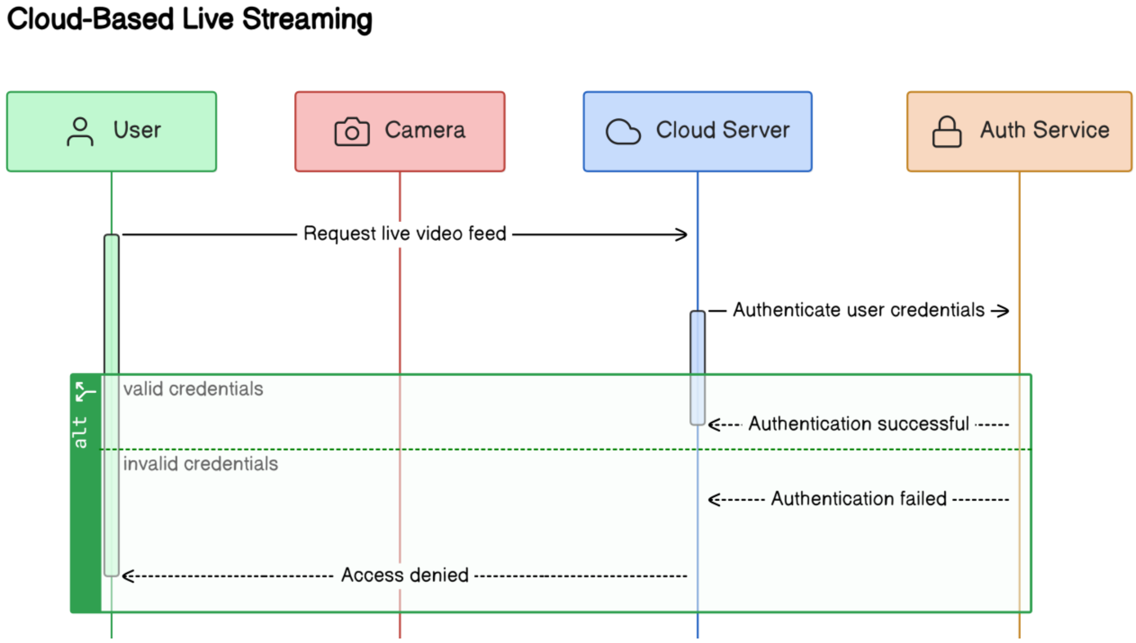


Figure 5 sequence diagrams for Cloud-Based Live Streaming

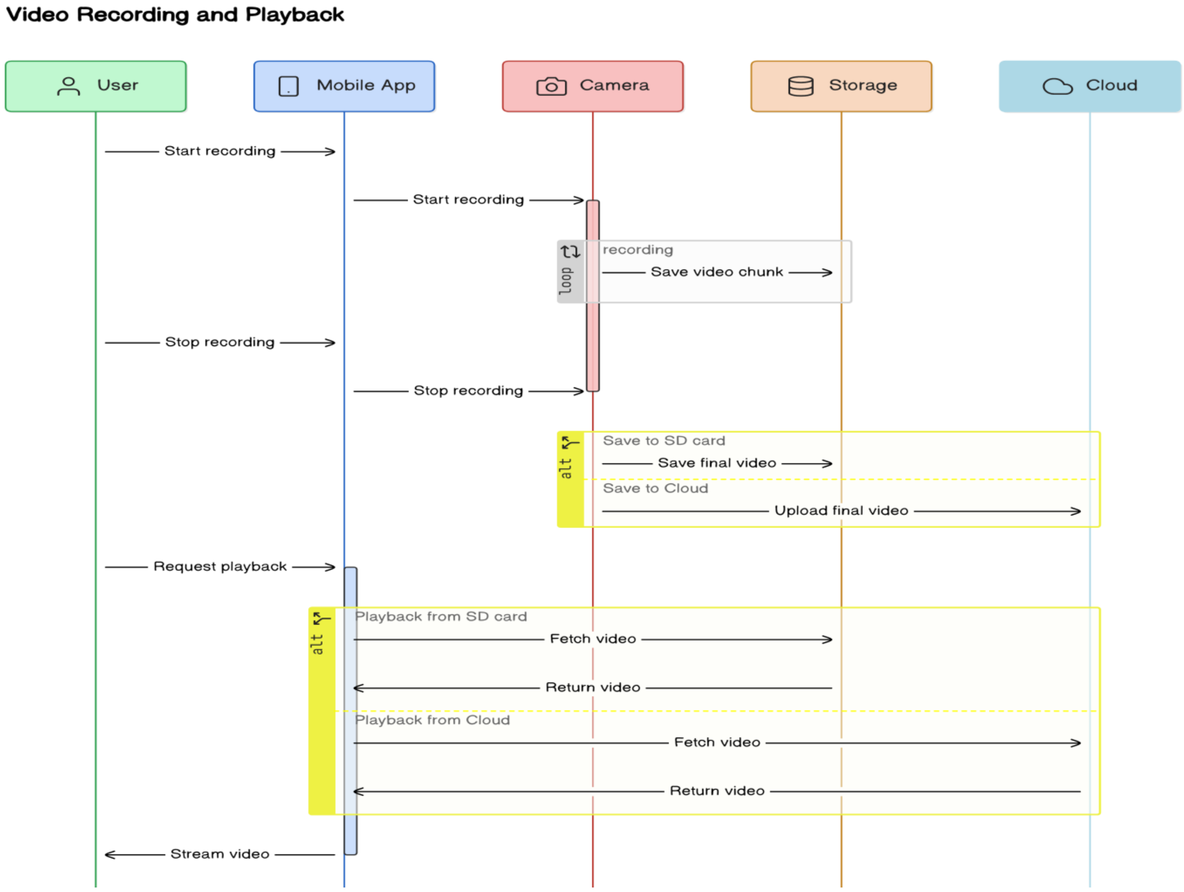


Figure 6 sequence diagrams Video Recording and Playback

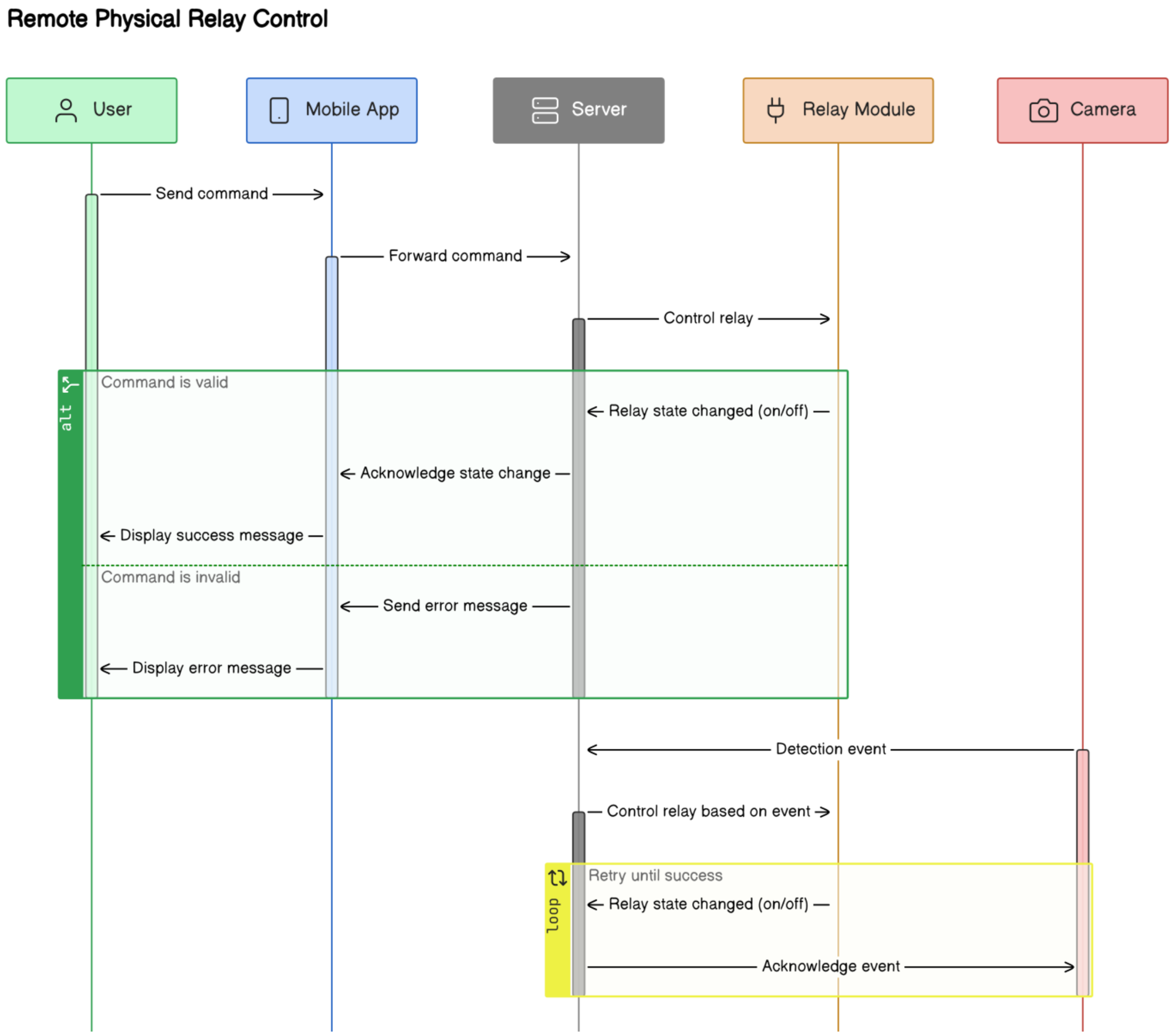


Figure 7 sequence diagrams for Remote Physical Relay Control

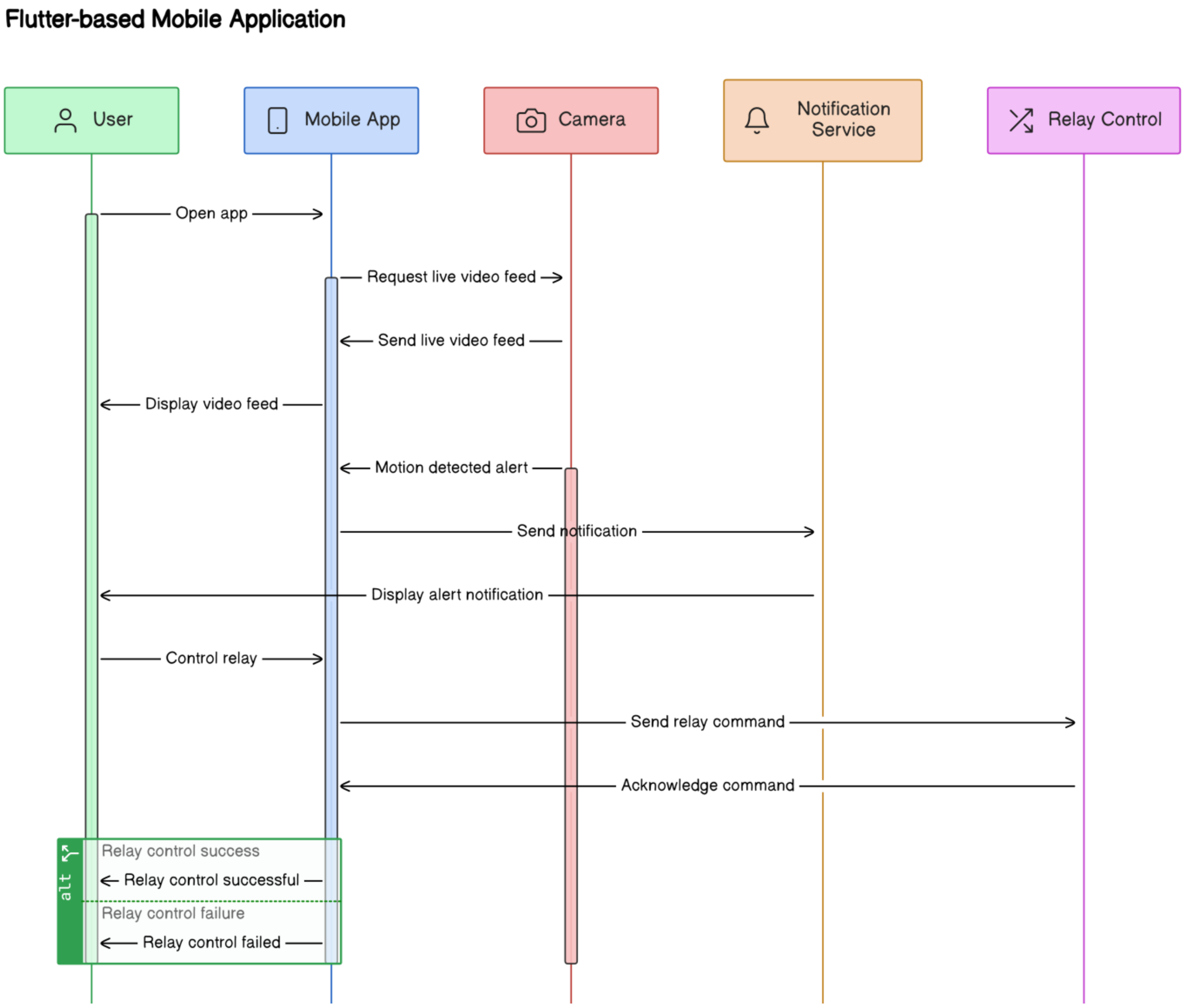


Figure 8 sequence diagrams for Flutter-based Mobile Application

Security Alerts and Notifications

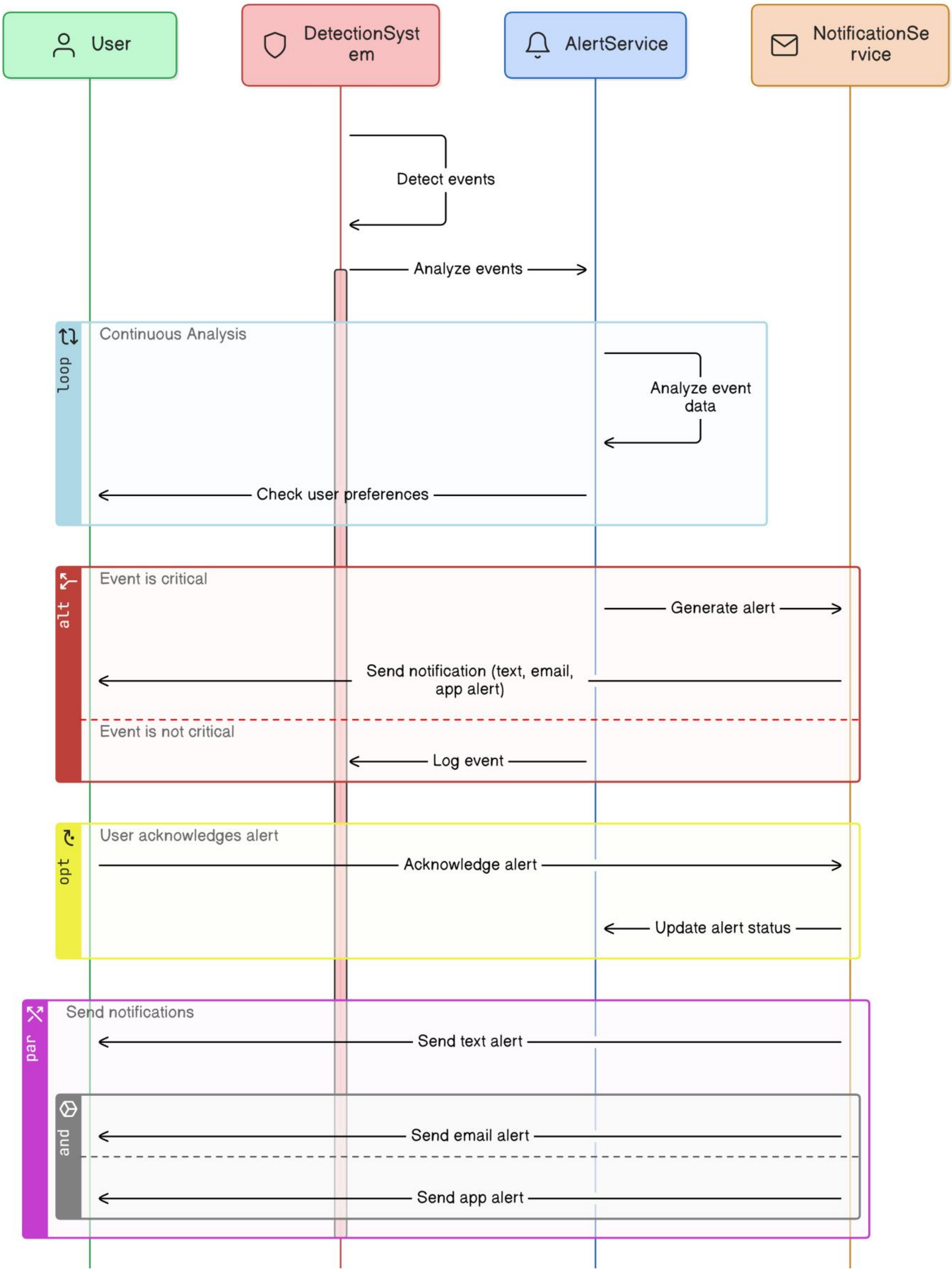


Figure 9 sequence diagrams for Security Alerts and Notifications

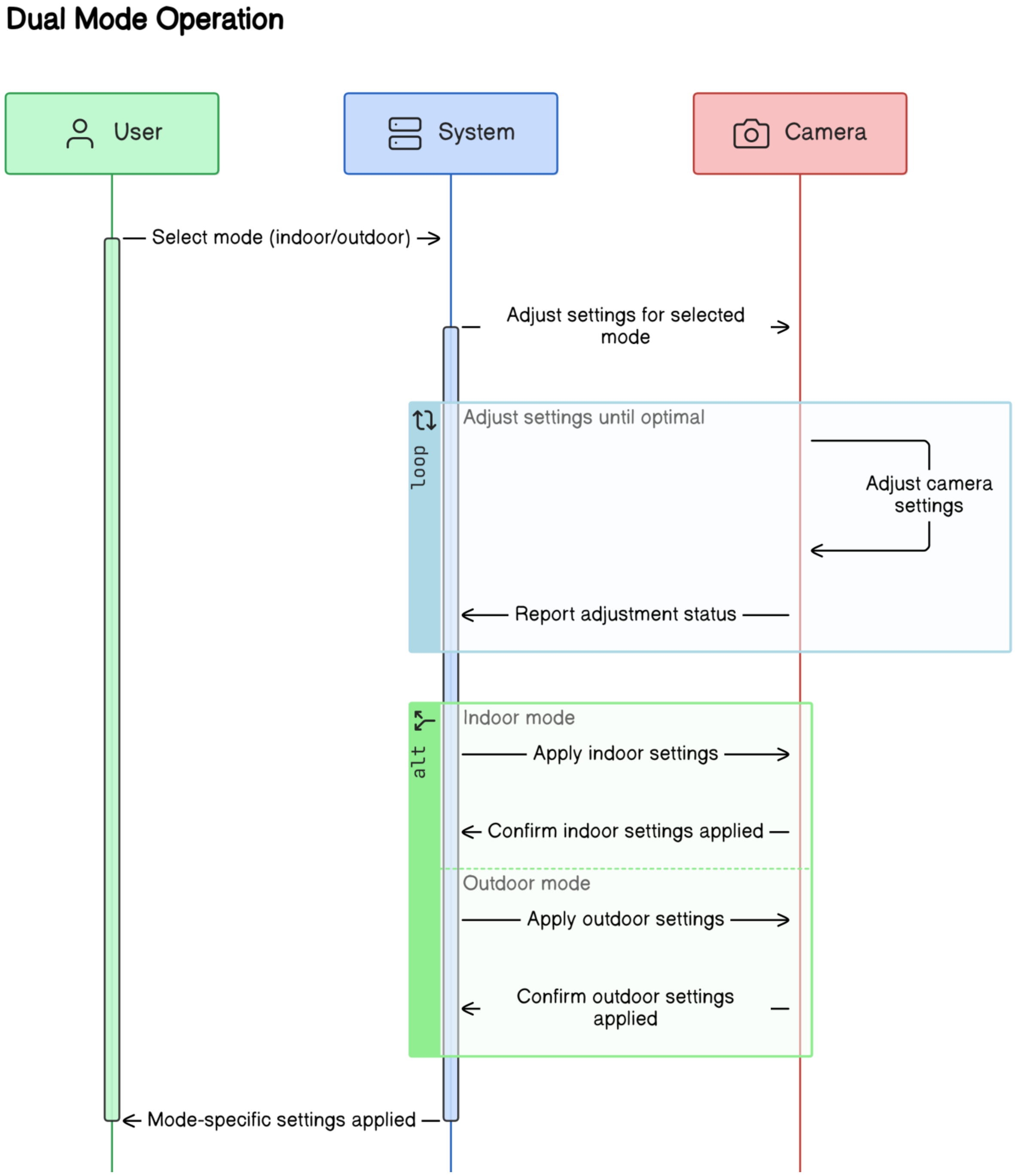

Dual Mode Operation
User
System
Camera
Select mode (indoor/outdoor)
Adjust settings for selected mode
loop
Adjust settings until optimal
Adjust camera settings
Report adjustment status
alt
Indoor mode
Apply indoor settings
Confirm indoor settings applied
Outdoor mode
Apply outdoor settings
Confirm outdoor settings applied
Mode-specific settings applied

Figure 10 sequence diagrams for Dual Mode Operation

2.4 Tools, libraries, special algorithms and implementation environment

ESP32-CAM WiFi IP Camera

- Hardware: ESP32-CAM Module (Espressif Systems)
  Reference: Espressif Official Documentation
- Firmware: Custom firmware
  Reference: ESP32-CAM Projects Guide
- **Libraries:**
  - ESP-IDF: Espressif IoT Development Framework
    Reference: ESP-IDF Documentation
  - Arduino IDE: Integrated Development Environment
    Reference: Arduino IDE

AI-Powered Object Detection

- Algorithm: YOLOv8 (You Only Look Once, version 8)
  Reference: YOLO Official Documentation
- **Libraries**
  - OpenCV: Computer vision library
    Reference: OpenCV Documentation
  - PyTorch: Deep learning framework
    Reference: PyTorch Documentation
  - Darknet: YOLO implementation
    Reference: Darknet GitHub
- **Tools**
  - Google Colab: Online notebook for running AI models
    Reference: Google Colab

Cloud-Based Live Streaming

- **Services:**
  - AWS (Amazon Web Services): Cloud infrastructure
    Reference: AWS Documentation
  - Firebase: Real-time database and authentication service
    Reference: Firebase Documentation
- **Libraries:**
  - FFmpeg: Multimedia framework for streaming
    Reference: FFmpeg Documentation
  - WebRTC: Real-time communication protocol
    Reference: WebRTC Documentation

Video Recording and Playback

- **Libraries**
  - FFmpeg: For video encoding and decoding
    Reference: FFmpeg Documentation
  - Local Storage: ESP32-CAM's SD card
    Reference: ESP32-CAM Storage Guide
- **Tools**
  - Node.js: Backend service development
    Reference: Node.js Documentation

Programmable Physical Relay Control

- Hardware: Relay modules
  Reference: Relay Module Documentation
- **Libraries**
  - ESP32 GPIO Library: For controlling GPIO pins
    Reference: ESP32 GPIO Documentation
- **Tools**
  - Arduino IDE:
    Reference: Arduino IDE

Flutter Mobile Application

- Framework: Flutter
  Reference: Flutter Documentation
- **Libraries**
  - Flutter Blue: Bluetooth communication
    Reference: Flutter Blue Documentation
  - HTTP: API interaction library
    Reference: Flutter HTTP Documentation
- **Tools**
  - Android Studio: IDE
    Reference: Android Studio Documentation
  - Visual Studio Code: IDE
    Reference: VS Code Documentation

Security Alerts and Notifications

- **Libraries**
  - Firebase Cloud Messaging (FCM): Push notifications
    Reference: FCM Documentation
- **Tools**
  - Python: Backend development
    Reference: Python Documentation

Implementation Environment

- **Development Environment**
    - Local: For ESP32-CAM development and testing.
    - Cloud: AWS, Goggle or another cloud provider for hosting live streaming and data services.
- **Testing Environment**
    - Simulated: Use virtual environments and emulators for mobile app testing.
    - Physical: Deploy cameras and test in real indoor and outdoor scenarios.
- **Version Control**
    - Git: For source code management.
    - GitHub or GitLab: For repository hosting and collaboration.

3 Interface Design

3.1 PACT (People, Activities, Contexts, Technologies) analysis of the system

People

The primary users of the "Intelligent Indoor/Outdoor Surveillance Camera with AI Detection and Programmable Relay Control" system are homeowners, security personnel, and business owners. These users require an easy-to-use interface for real-time monitoring and control over surveillance feeds. Since the system is operated through a mobile app, the user interface must cater to different levels of technical proficiency. While homeowners may use the system for general security, security personnel might need more advanced functionalities like detailed notifications and remote device control.

Activities

The main activities include:

- **Monitoring live video feeds** from indoor and outdoor cameras.
- **Controlling external devices** such as alarms or lights through relay systems.
- **Receiving and managing security alerts**, such as notifications triggered by AI object detection.

Contexts

The system operates in diverse environments such as homes, offices, warehouses. It functions in both indoor and outdoor settings and must be capable of handling different lighting and weather conditions. The mobile app provides users with remote access to live video streams,security alerts, and controls for external devices, even when they are away from the monitored site. This makes the system ideal for situations where users need real-time updates and control over the surveillance environment.

Technologies

The core technologies involved in the system include:

- **ESP32-CAM module** for video capture and relay control integration.
- **YOLOv8** for AI-powered real-time object detection and classification.
- **Cloud-based services** for live streaming and data storage to ensure remote access and security.
- **Flutter-based mobile app** that serves as the user interface for video monitoring, receiving alerts, and controlling connected devices. The system uses Wi-Fi, RTSP streaming protocol, and programmable relays to achieve seamless hardware and software integration.

3.2 Interfaces (software/hardware) of the system

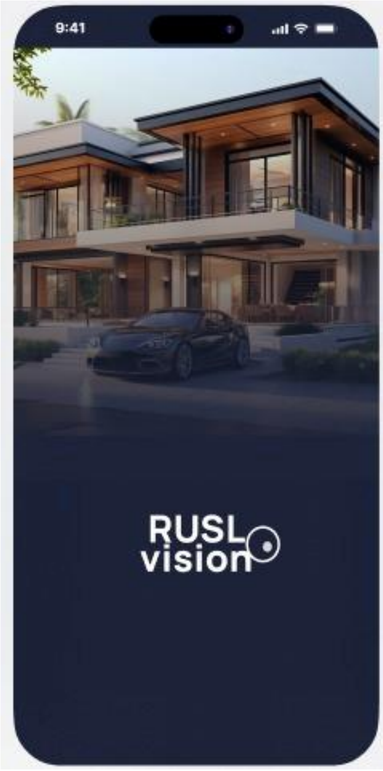
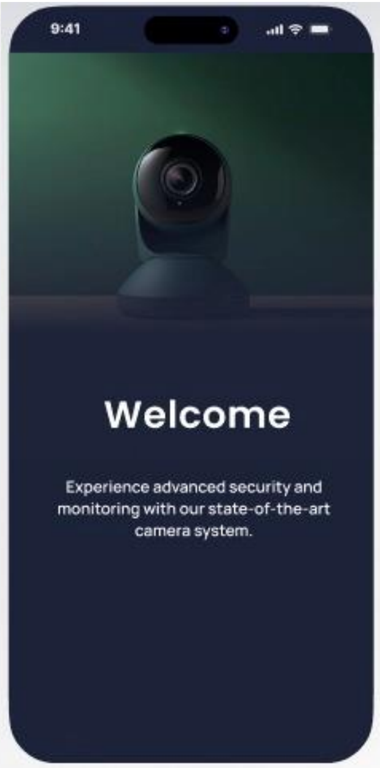


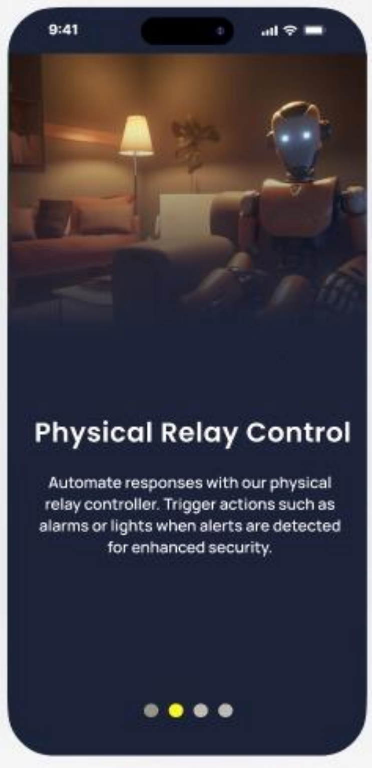
Figure 11 Startup interfaces



Welcome

Experience advanced security and monitoring with our state-of-the-art camera system.

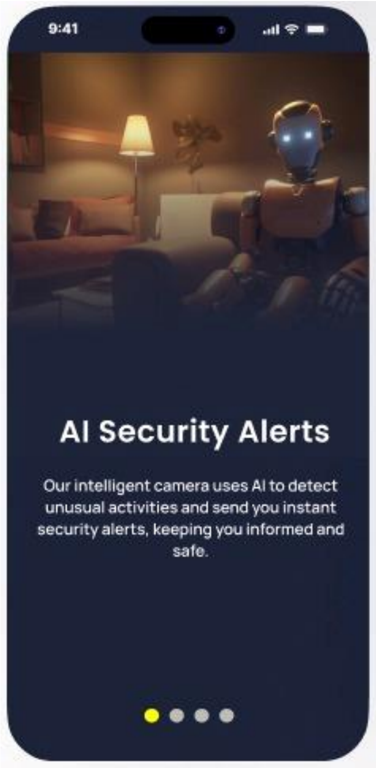
Figure 12 Startup interfaces



Physical Relay Control

Automate responses with our physical relay controller. Trigger actions such as alarms or lights when alerts are detected for enhanced security.

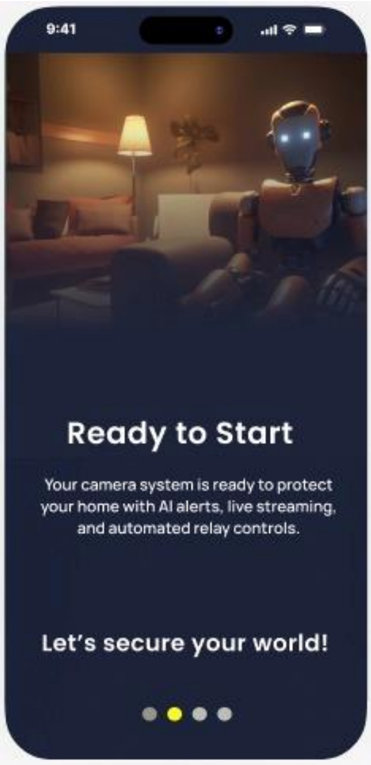
Figure 13 Startup interfaces



AI Security Alerts

Our intelligent camera uses AI to detect unusual activities and send you instant security alerts, keeping you informed and safe.

Figure 14 Startup interfaces

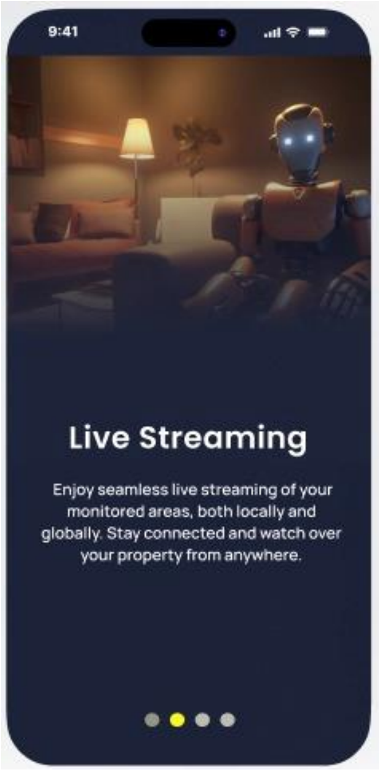


Ready to Start

Your camera system is ready to protect your home with AI alerts, live streaming, and automated relay controls.

Let's secure your world!

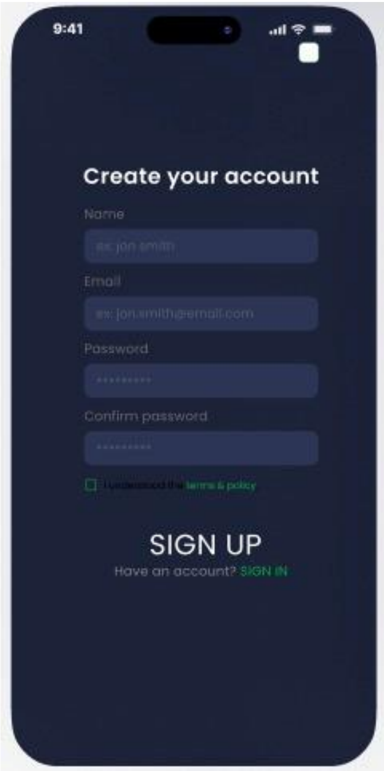
Figure 15 Startup interfaces



Live Streaming

Enjoy seamless live streaming of your monitored areas, both locally and globally. Stay connected and watch over your property from anywhere.

Figure 16 Startup interfaces



Create your account

Name

ex: jon smith

Email

ex: jon.smith@email.com

Password

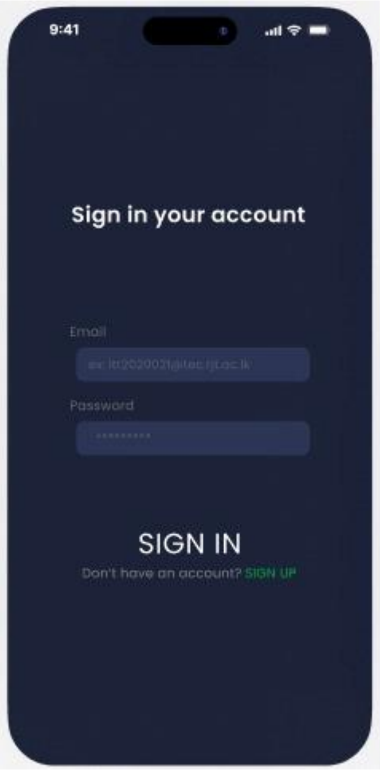
****

Confirm password

****

□ I understood the terms & policy

Have an account? SIGN IN



Sign in your account

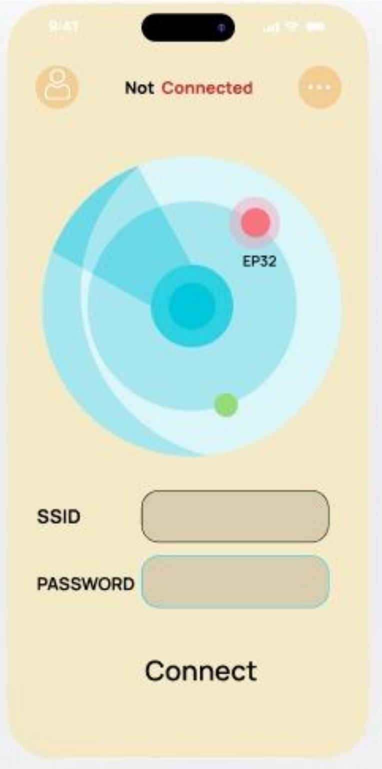
Email

ex: itr2020021@tec.rjt.ac.lk

Password

****

Don't have an account? SIGN UP



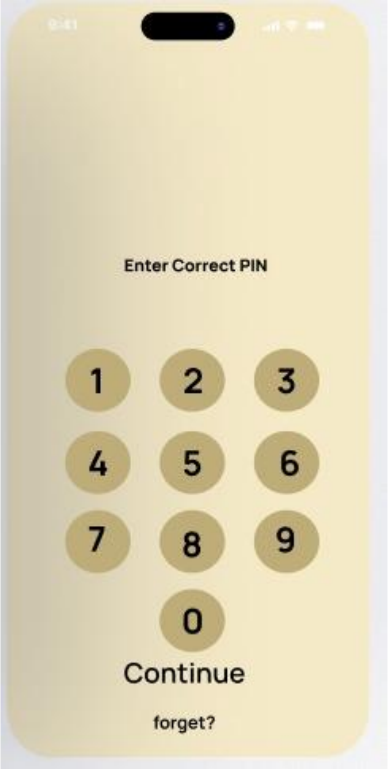
SSID

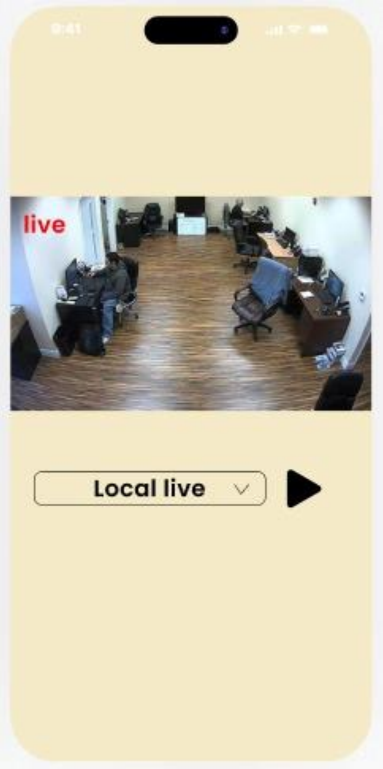
PASSWORD

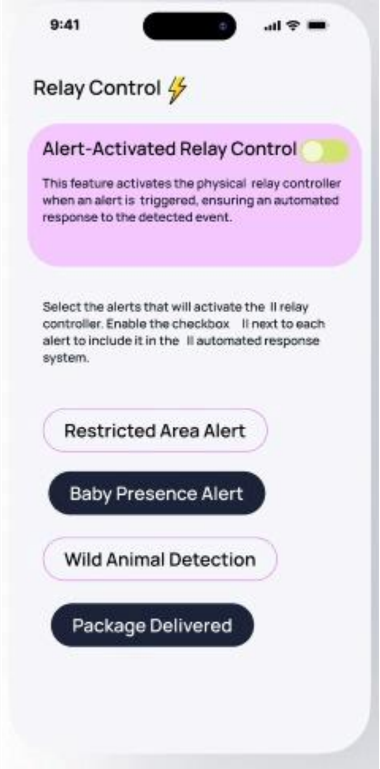
Figure 17 Signup interfaces

Figure 18 Sign in interfaces

Figure 19 connect camara interfaces







Relay Control ⚡

Alert-Activated Relay Control

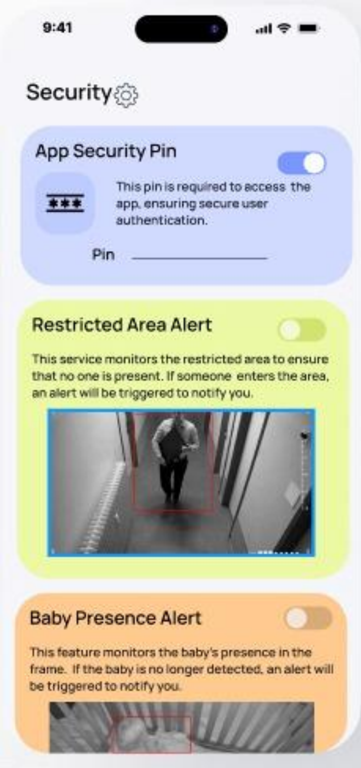
This feature activates the physical relay controller when an alert is triggered, ensuring an automated response to the detected event.

Select the alerts that will activate the II relay controller. Enable the checkbox II next to each alert to include it in the II automated response system.

Figure 20 PIN interfaces

Figure 21 Home interfaces

Figure 22 Relay Control interfaces

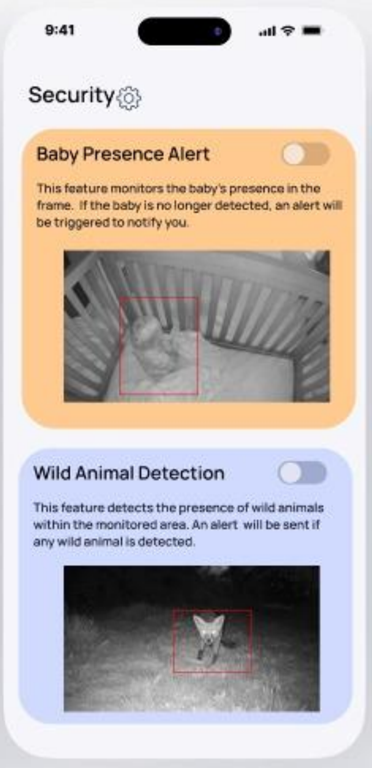


This pin is required to access the app, ensuring secure user authentication.

This service monitors the restricted area to ensure that no one is present. If someone enters the area, an alert will be triggered to notify you.

This feature monitors the baby's presence in the frame. If the baby is no longer detected, an alert will be triggered to notify you.

Figure 23 Security interfaces



This feature monitors the baby's presence in the frame. If the baby is no longer detected, an alert will be triggered to notify you.

This feature detects the presence of wild animals within the monitored area. An alert will be sent if any wild animal is detected.

Figure 24 Security interfaces

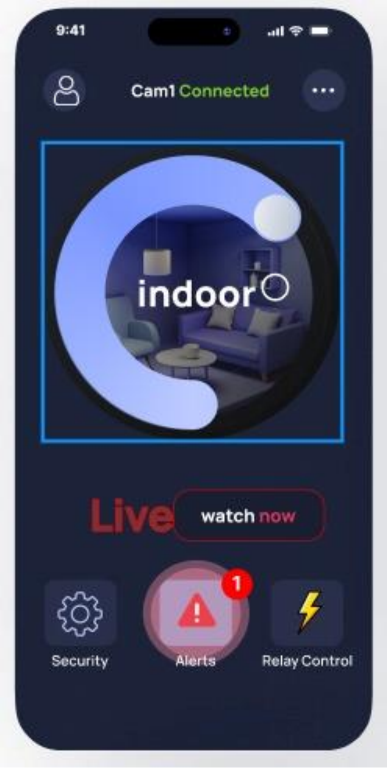
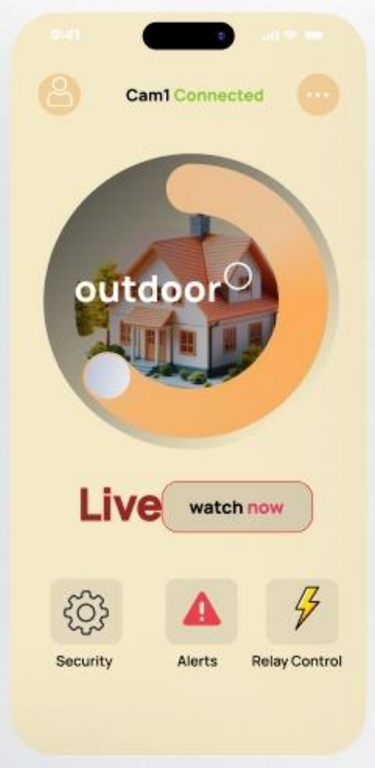


Figure 25 indoor Streaming interfaces



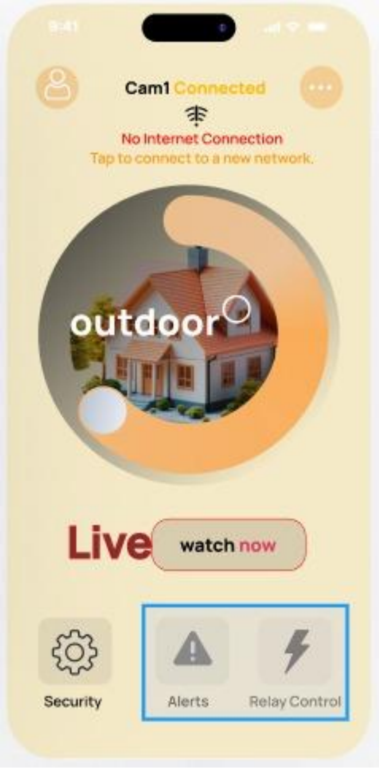


Figure 26 outdoor Streaming interfaces *Figure 27 outdoor Streaming interfaces*

3.3 Design tools, techniques, templates

Design Tools

- **Figma/Adobe XD**: These tools were used for designing the user interface (UI) and user experience (UX) of the mobile application. They allowed for the creation of wireframes and prototypes before final implementation.

- **Arduino IDE**: Used to develop and program the ESP32-CAM for its core functionalities like live streaming, object detection, and relay control.

- **PyCharm/VS Code**: These development environments were used for programming AI algorithms, including YOLOv8 for object detection, as well as managing backend services.

Techniques

- **Wireframing and Prototyping**: Early-stage designs for the mobile app UI were created using wireframing techniques. Prototypes were developed to test usability and layout before finalizing the design.

- **Test-Driven Development (TDD)**: The AI detection system and other core features were developed using TDD to ensure the system functions as expected with a high level of accuracy.

Templates

- **Mobile App UI Templates**: Predefined UI components like buttons, sliders, and video panels were utilized to accelerate app development. These templates ensured consistency and efficiency in the design process.

- **Backend API Templates**: Standardized API templates were used to link the mobile app with cloud services and video streaming functionalities. These templates ensured a smooth integration between the front-end and back-end services.

4 Data Management

4.1 Design tools, techniques

For the design and implementation of the database in our surveillance system, we employed several tools and techniques to ensure efficient management of data related to users, cameras, recordings, and streams.

- **Database Management System (DBMS): We** have opted for MySQL as our DBMS due to its reliability, scalability, and strong support for handling relational data.
- **Entity-Relationship (ER) Diagram:** We utilized draw.io to create the conceptual ER diagrams. This tool allowed us to visualize the relationships between the different entities (User, Camera, Recording, Stream) and their attributes.
- **UML Diagrams:** For logical and physical database designs, we applied UML (Unified Modeling Language) to create comprehensive models. These diagrams help in understanding the cardinalities, primary and foreign keys, and attribute constraints.
- **Normalization:** We used normalization techniques to ensure that the database is free from redundancy and update anomalies. The database is normalized up to the third normal form (3NF), ensuring minimal data duplication and optimal storage efficiency.
- **SQL Queries:** Structured Query Language (SQL) is used to manage the interaction with the database. It includes queries for creating tables, defining constraints, and inserting data.
- **Cloud Integration:** The database is integrated with a cloud-based infrastructure to support remote access, ensuring that users can retrieve their surveillance data from any location securely.

4.2 Conceptual database design

The conceptual design of the database was initially developed using an Entity-Relationship (ER) diagram. This stage defines the relationships and data flow between the four primary entities in the system: User, Camera, Recording and Stream. Each entity contains various attributes, and their relationships have been clearly outlined.

- User Entity
  The User entity contains the following attributes.
  - UserID (Primary Key)
  - Username
  - Password
  - Email
- Camera Entity
  The Camera entity manages information about the surveillance cameras, with these attributes.
  - CameraID (Primary Key)
  - IPAddress
- Recording Entity
  The Recording entity stores details of recorded video files.
  - RecordingID (Primary Key)
  - Start_Time
  - End_Time
  - Storage_Location

- Stream Entity
  The Stream entity represents live streams captured by the cameras.
  - StreamID (Primary Key)
  - Start_Time
  - End_Time
  - Cloud_URL
- Relationships
  - A User can control multiple Cameras.
  - A Camera can have multiple associated Recordings and Streams.
  - A Camera can have one associated Streams.
  - Recording and Stream entities are related through the Camera entity.
- Constraints
  - Each User is uniquely identified by UserID.
  - Each Camera must be assigned a uniquely identified CameraID.
  - Each Recording must be assigned a uniquely identified RecordingID.
  - Each Stream must be assigned a uniquely identified StreamID.

4.3 Logical database design and schema refinement

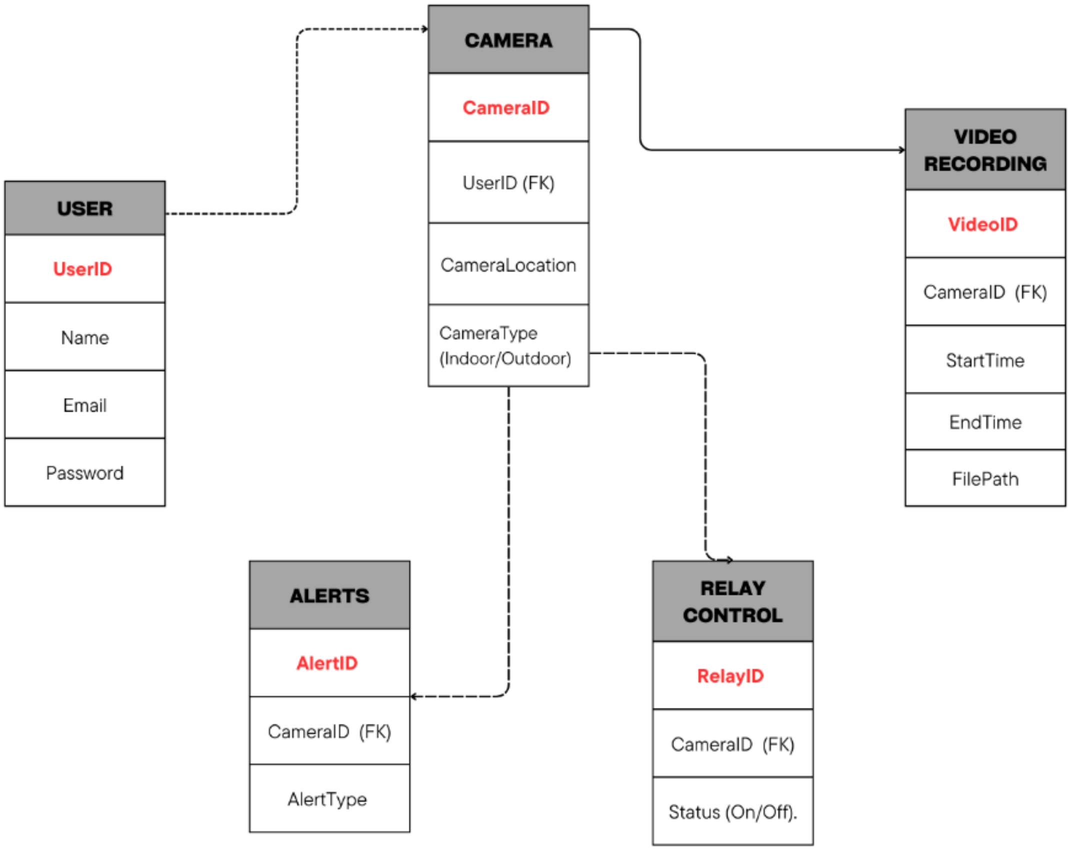


Figure 28 refined-database schema

4.4 Physical database design

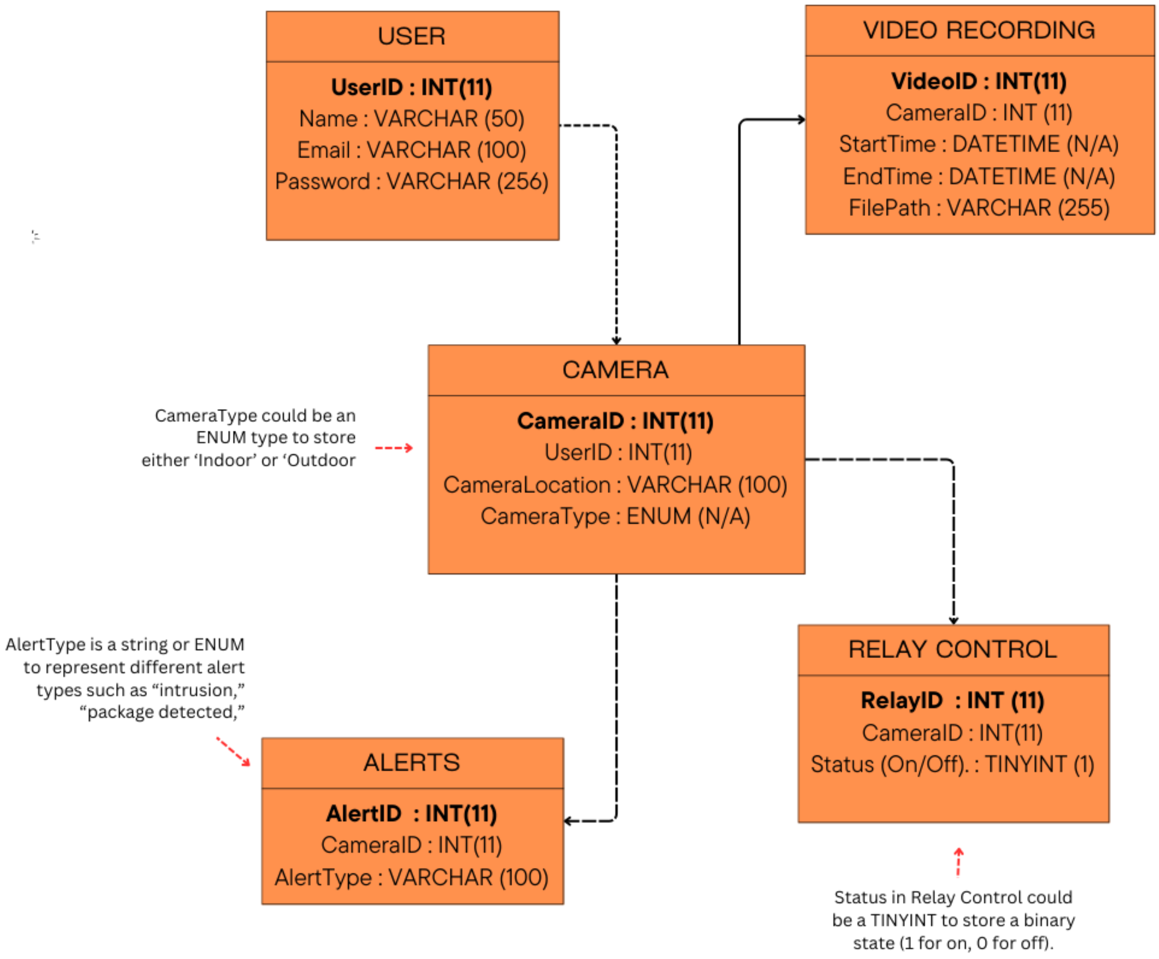

USER
UserID : INT(11)
Name : VARCHAR (50)
Email : VARCHAR (100)
Password : VARCHAR (256)
VIDEO RECORDING
VideoID : INT(11)
CameraID : INT (11)
StartTime : DATETIME (N/A)
EndTime : DATETIME (N/A)
FilePath : VARCHAR (255)
CameraType could be an ENUM type to store either 'Indoor' or 'Outdoor
CAMERA
CameraID : INT(11)
UserID : INT(11)
CameraLocation : VARCHAR (100)
CameraType : ENUM (N/A)
AlertType is a string or ENUM to represent different alert types such as "intrusion," "package detected,"
RELAY CONTROL
RelayID : INT (11)
CameraID : INT(11)
Status (On/Off). : TINYINT (1)
ALERTS
AlertID : INT(11)
CameraID : INT(11)
AlertType : VARCHAR (100)
Status in Relay Control could be a TINYINT to store a binary state (1 for on, 0 for off).

Figure 29 Physical ER diagram

Constraints: Added constraints to enforce data integrity:

- **NOT NULL:** Important fields like Email, Password, CameraID, etc., are now marked as NOT NULL to ensure these fields are always filled.
- **UNIQUE:** Applied to Email to ensure that no two users can have the same email.

5 Hardware Design

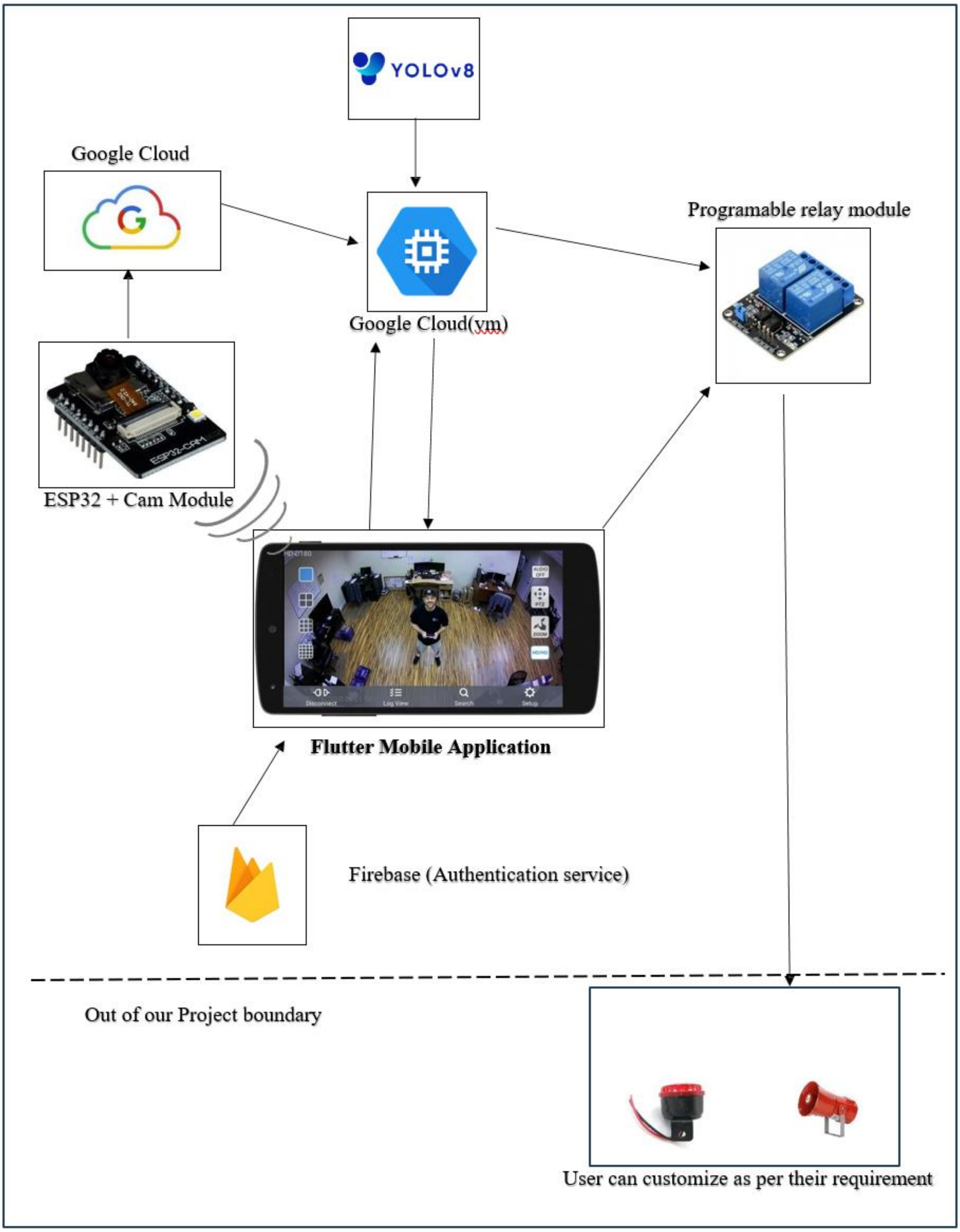

YOLOv8
Google Cloud
Google Cloud(vm)
Programable relay module
ESP32 + Cam Module
Flutter Mobile Application
Firebase (Authentication service)
Out of our Project boundary
User can customize as per their requirement

Figure 30 architecture diagrams

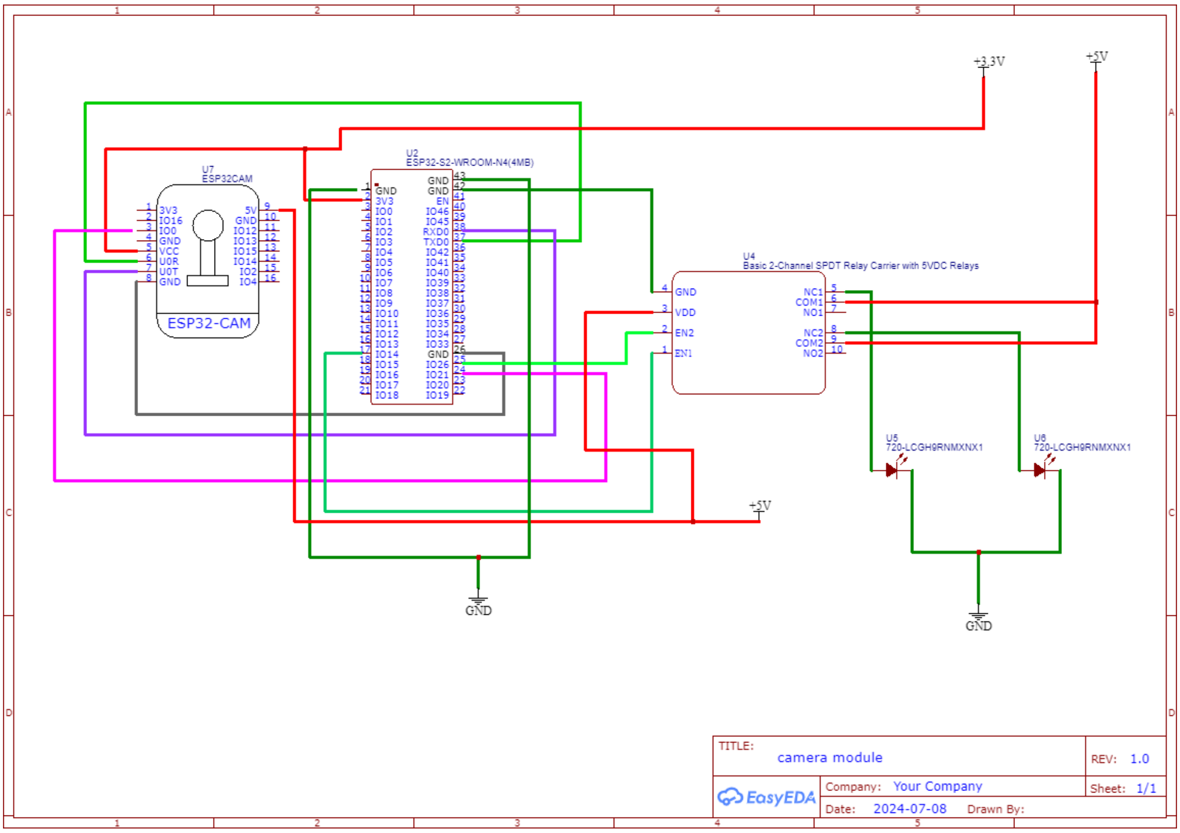


Figure 31 Hardware Design

6 Recommendation of supervisor(s) on the document

(This section should be filled by the supervisor(s)).

Comments (if any):

I/We certify that, the student engaged continuously with me in developing the proposal and, I am confident that they are adequately competent to defend this viva.

Signature(s) of Supervisor(s):

Date:

7 Viva presentation assessment team

(This section should be filled by the department)

Date of viva presentation: --

Panel members	Name	Department / Institute
Chair		
Member		
Member		
Member		
Member		

8 Comments of the assessment team on viva presentation

(This should be filled by the chair of the assessment panel. In case of revision or fail, needed revision or reasons to fail the viva presentation should be mentioned here)

Result of the viva presentation	**Excellent / Good / Pass with revisions / Fail**
Score	
Signature of the panel chair	
Date	