



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

(Leave this page blank)

Details of the Project


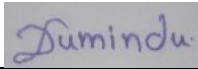
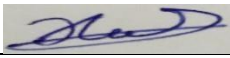
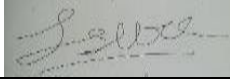
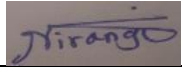


Project Title : Intelligent Indoor OR Outdoor Surveillance Camera with AI
Detection and Programmable Relay Control

Group Number : Group 01

Group Name : Tech Titans

Submission Date : 2024/09/18

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 |  |
| H.L.I.N. Liyanaarachchi | ITT/2020/128 | 1390 |  |
| P.W.D.I.M. Rodrigo | ITT/2020/091 | 1357 |  |
| B.K. Bandaranayake | ITT/2020/121 | 1383 |  |
| K.R.N. Perera | ITT/2020/072 | 1343 |  |
| 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

Guideline to fill the document

1. SDS document should elaborate the details with respect to the Design phase of the project.
2. Your design decisions should tally with the Software Requirements Specification (SRS) which you have already submitted in the previous stage.
3. Software Design Specification document should be prepared using a word processor, clear and coherent.
4. The content of each section in this document should be in detail and make the focal points stand out.
5. A student may have more than one supervisor. In such case, the details of all supervisors should be included on the document.
6. The student can attach additional pages.
7. The student must follow the IEEE format to make the reference list.
8. This document should be printed on both sides of A4 papers (do not print with colors).
9. Each figure/diagram should be numbered along with a caption. All figures/diagrams must be referred within the textual content. The figure number and caption must be placed at the bottom of the figure/diagram.
10. Each table should be numbered along with a caption. All tables must be referred within the textual content. The table number and caption must be placed at the top of the table.

Submission Guidelines for the document

1. The student must submit a soft copy of finalized document to the Moodle, as an electronic book version (.pdf) on or before the submission date.
2. Meanwhile, a hard copy of the document which is recommended and approved by the supervisor(s) (with signature(s)), must be submitted to the department before the viva presentation date.
3. The student must submit the second version of the document to the Moodle, after corrections made based on the feedbacks and comments of viva panel as an electronic book version (.pdf). In this case, recommendation and approval page of supervisor(s) and comments/ approval page of viva panel could be scanned and attached at the end of the document in order to make the final document in electronic book version (.pdf).

Table of Contents

| | | |
|-----|---|----|
| 1 | Introduction..... | 1 |
| 1.1 | Background of the project..... | 1 |
| 1.2 | Purpose and significance of the project | 1 |
| 1.3 | Scope of the project | 1 |
| 1.4 | Objectives of the Project | 2 |
| 1.5 | System design approach..... | 2 |
| 2 | Architectural Design | 4 |
| 2.1 | System Architecture..... | 4 |
| 2.2 | Component design | 5 |
| 2.3 | Processes and interaction design..... | 6 |
| 2.4 | Tools, libraries, special algorithms and implementation environment | 12 |
| 3 | Interface Design | 15 |
| 3.1 | PACT (People, Activities, Contexts, Technologies) analysis of the system..... | 15 |
| 3.2 | Interfaces (software/hardware) of the system | 16 |
| 3.3 | Design tools, techniques, templates | 19 |
| 4 | Data Management | 20 |
| 4.1 | Design tools, techniques | 20 |
| 4.2 | Conceptual database design | 20 |
| 4.3 | Logical database design and schema refinement | 21 |
| 4.4 | Physical database design..... | 22 |
| 5 | Hardware Design | 23 |
| 6 | Recommendation of supervisor(s) on the document..... | 25 |
| 7 | Viva presentation assessment team..... | 26 |
| 8 | Comments of the assessment team on viva presentation | 26 |

List of Figures

| | |
|--|----|
| Figure 1 block diagrams..... | 4 |
| Figure 2 class diagram | 5 |
| Figure 3 sequence diagrams | 6 |
| Figure 4 sequence diagrams | 7 |
| Figure 5 sequence diagrams | 7 |
| Figure 6 sequence diagrams | 8 |
| Figure 7 sequence diagrams | 9 |
| Figure 8 sequence diagrams | 10 |
| Figure 9 sequence diagrams | 11 |
| Figure 10 interfaces..... | |
| Figure 11 interfaces..... | |
| Figure 12 interfaces..... | 16 |
| Figure 13 interfaces..... | |
| Figure 14 interfaces..... | |
| Figure 15 interfaces..... | 16 |
| Figure 16 interfaces..... | |
| Figure 17 interfaces..... | |
| Figure 18 interfaces..... | 17 |
| Figure 19 interfaces..... | |
| Figure 20 interface s | |
| Figure 21 interfaces..... | 17 |
| Figure 22 interfaces..... | |
| Figure 23 interfaces..... | |
| Figure 24 interfaces..... | 18 |
| Figure 25 interfaces..... | |
| Figure 26 interfaces..... | 18 |
| Figure 27 refined-database schema..... | 21 |
| Figure 28 Physical ER diagram | 22 |
| Figure 29 architecture diagrams..... | 23 |
| Figure 30 Hardware Design | 24 |

List of Tables

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:

1. **ESP32-CAM WIFI IP Camera:** Design and setup of the ESP32-CAM module, integrating a 2MP camera for video recording.
2. **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.
3. **Cloud-Based Live Streaming:** Development of a secure cloud-based solution for streaming live video feeds.
4. **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.
5. **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.

6. **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.
7. **Security Alerts and Notifications:** Intelligent mechanisms for analyzing events and providing context-aware alerts, customizable by users.

1.4 Objectives of the Project

1. **Developing the ESP32-CAM WIFI IP Camera** to integrate seamlessly with live streaming, recording, and relay control functionalities of the surveillance system.
2. **Integrating AI-Powered Object Detection** using YOLOv8 for accurate identification and classification of objects in real-time.
3. **Enable Cloud-Based Live Streaming** for remote access to the surveillance camera's live video feed with robust security measures.
4. **Implementing Video Recording Functionality** allowing users to start and stop recordings from the mobile app.
5. **Implementing Remote Physical Relay Control** enabling users to manage external devices (e.g., lights, alarms) remotely via the mobile app.
6. **Developing a Flutter-based Mobile Application** with user-friendly interfaces for monitoring live video, receiving security alerts, and controlling connected devices.
7. **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

For our "Intelligent Indoor/Outdoor Surveillance Camera with AI Detection and Programmable Relay Control" project, we the design **architecture of your system, you** might use:

- **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:

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

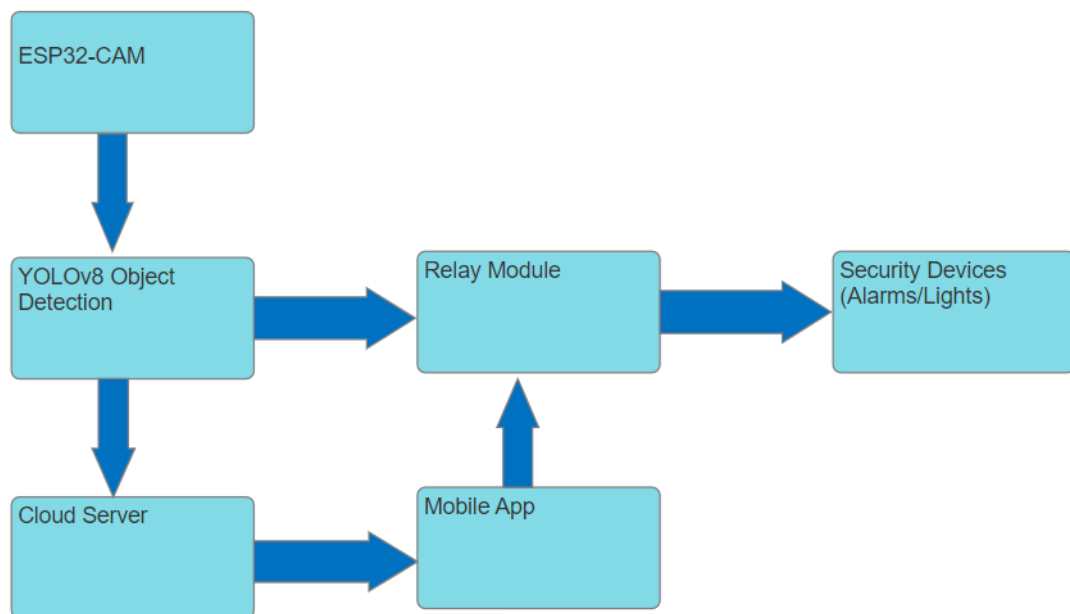


Figure 1 block diagrams

2.2 Component design

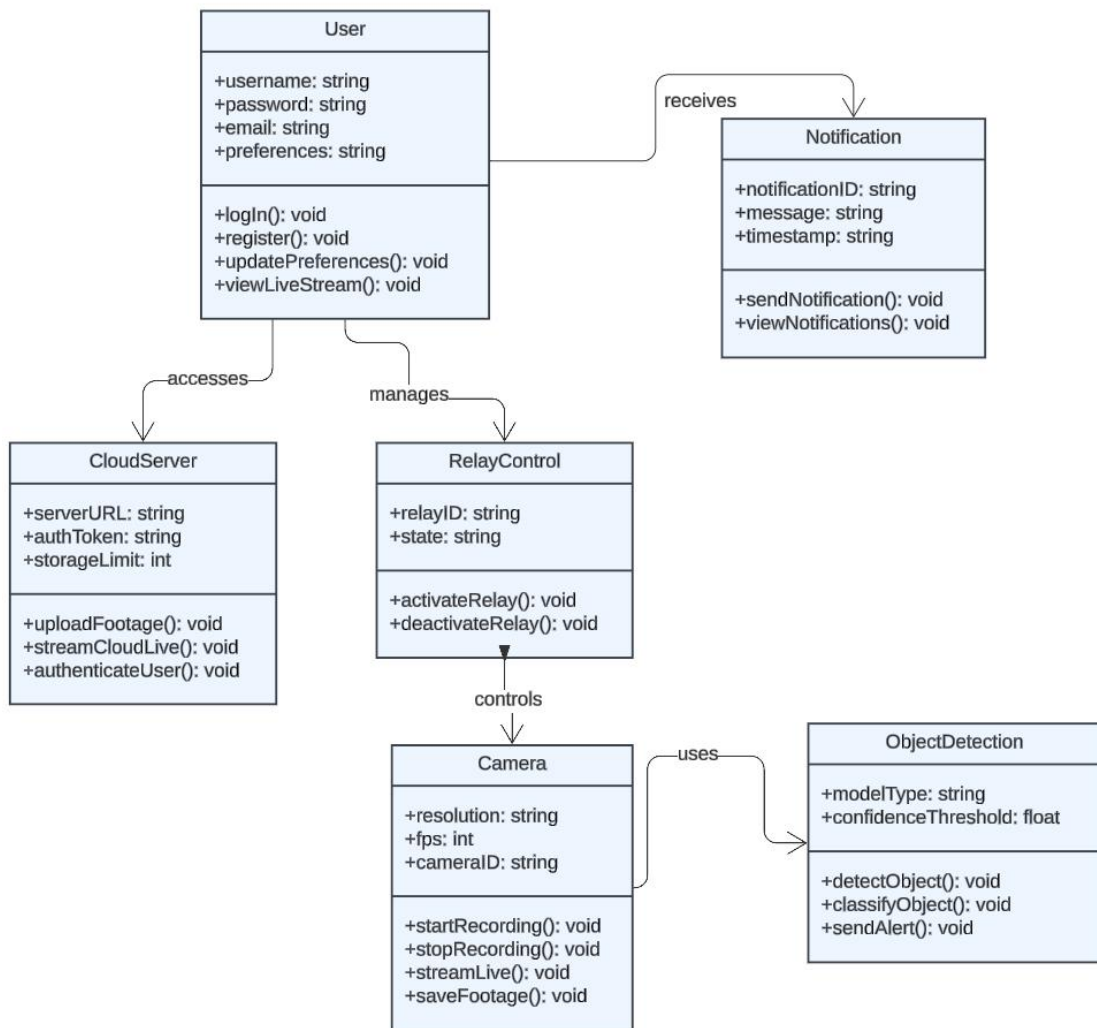


Figure 2 class diagram

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

AI Object Detection Process

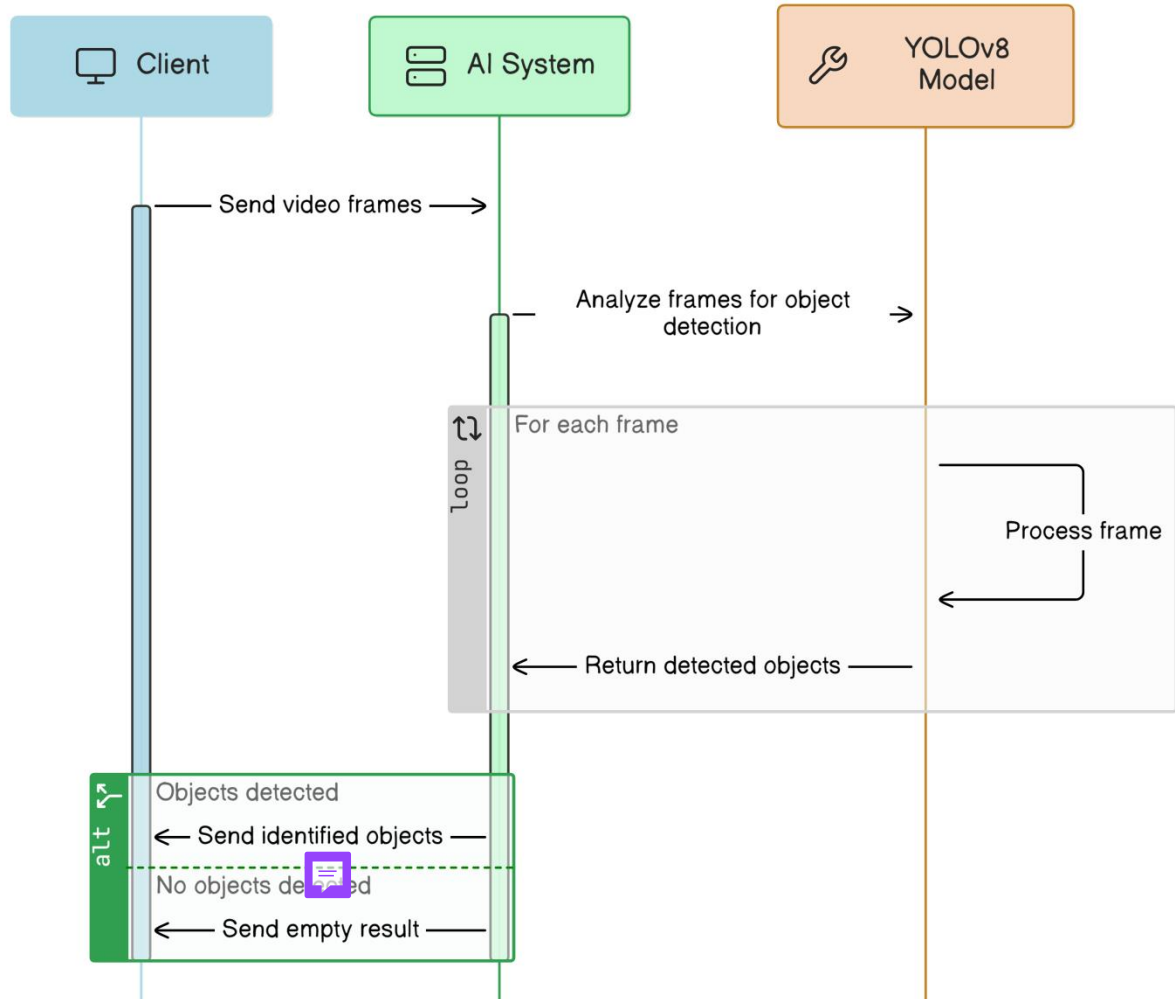


Figure 3 sequence diagram

Cloud-Based Live Streaming

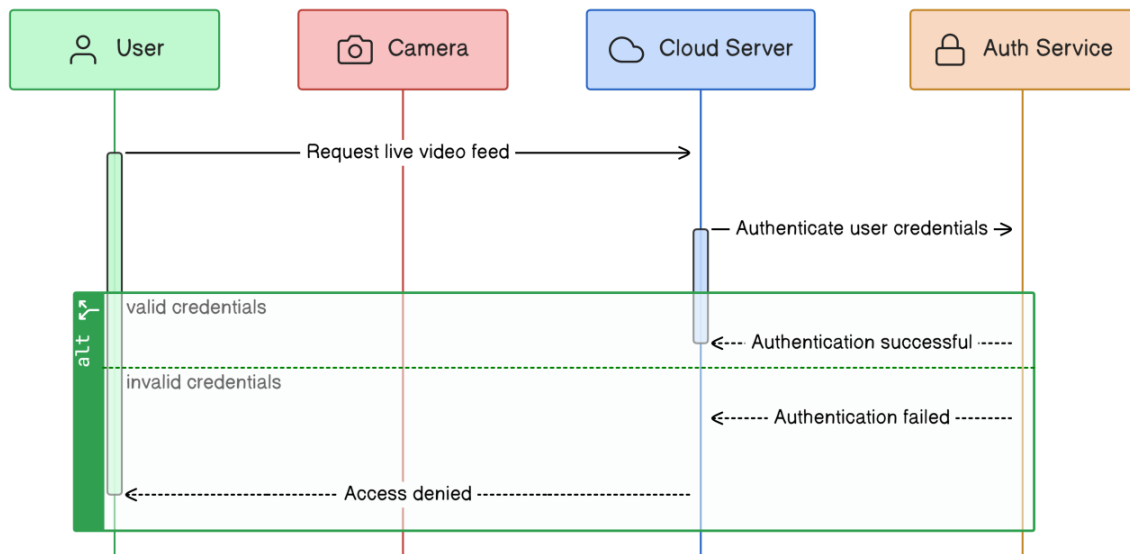


Figure 4 sequence diagrams

Video Recording and Playback

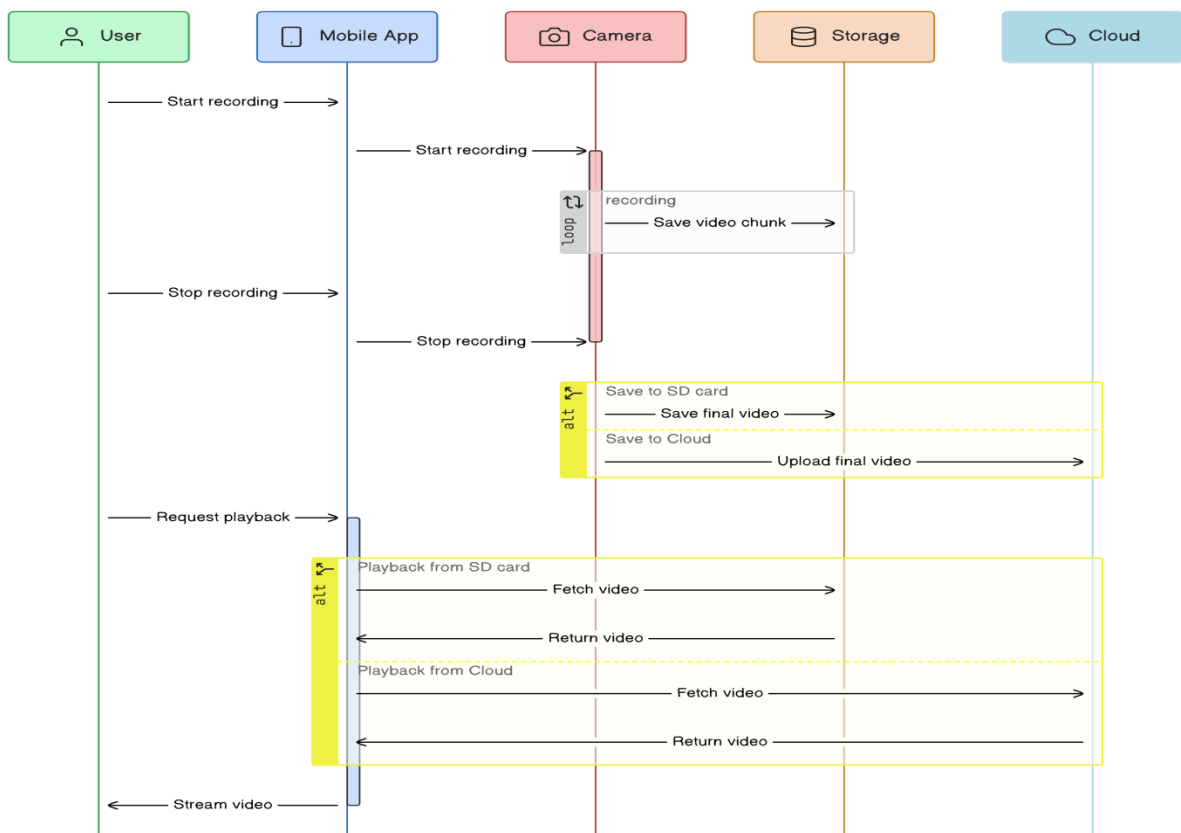


Figure 5 sequence diagrams

Remote Physical Relay Control

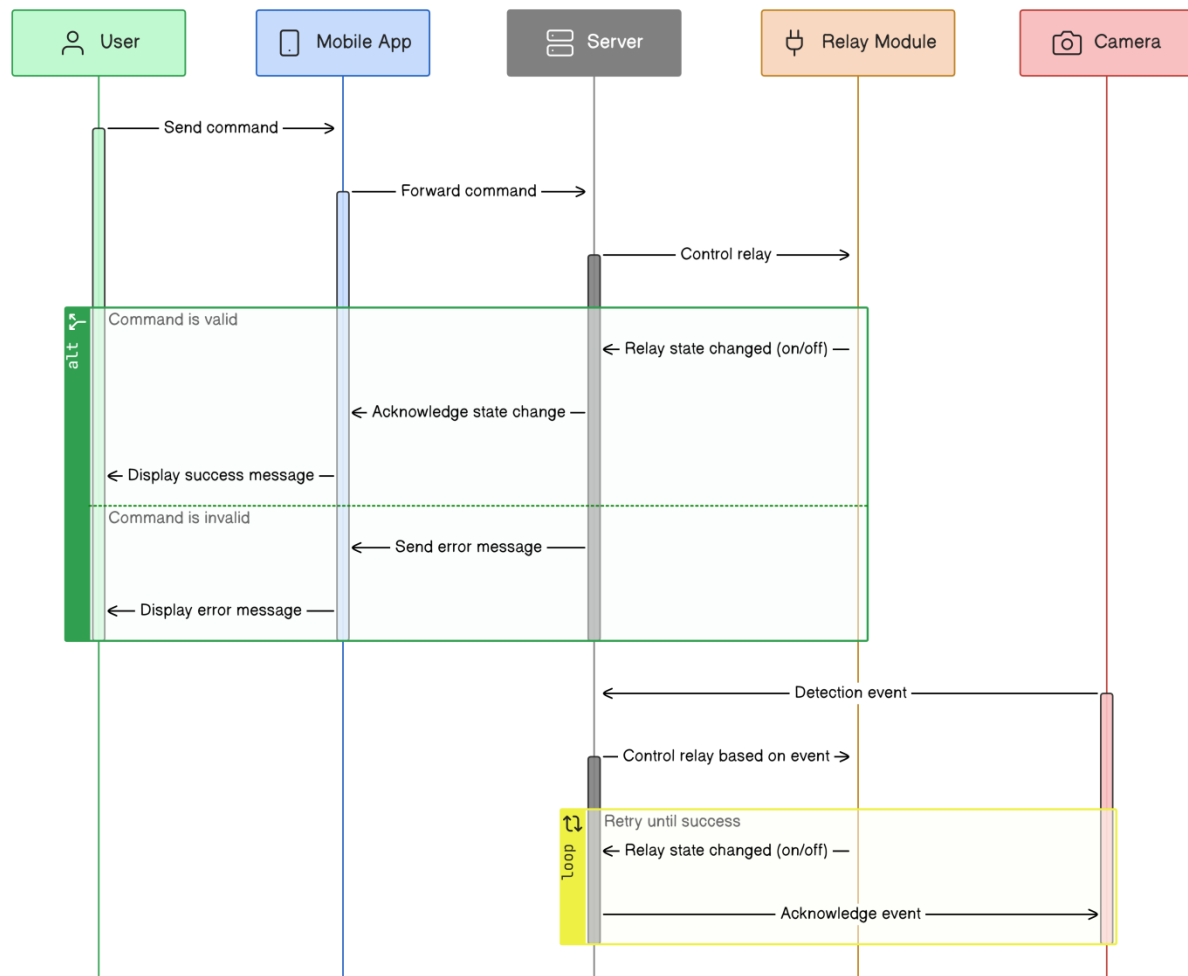


Figure 6 sequence diagrams

Flutter-based Mobile Application

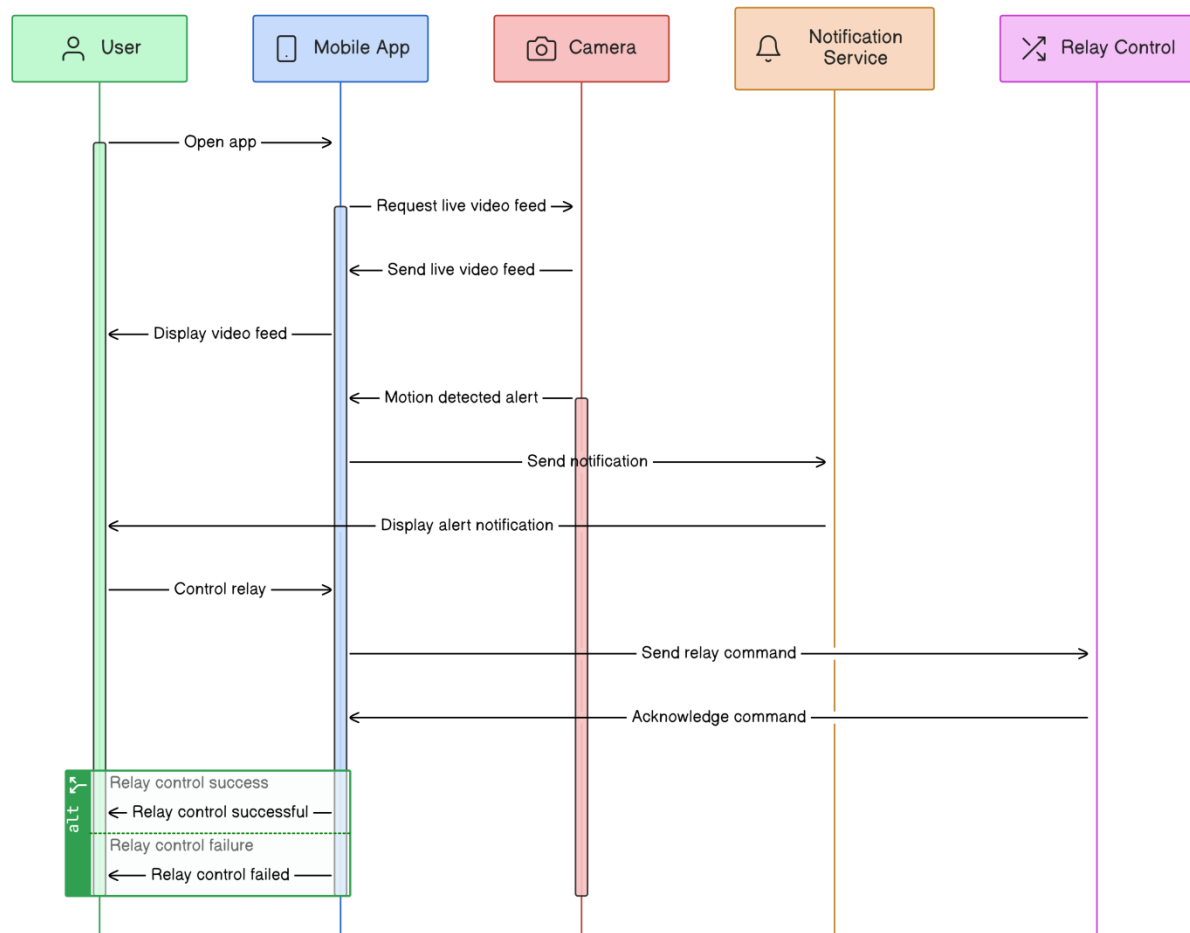


Figure 7 sequence diagrams

Security Alerts and Notifications

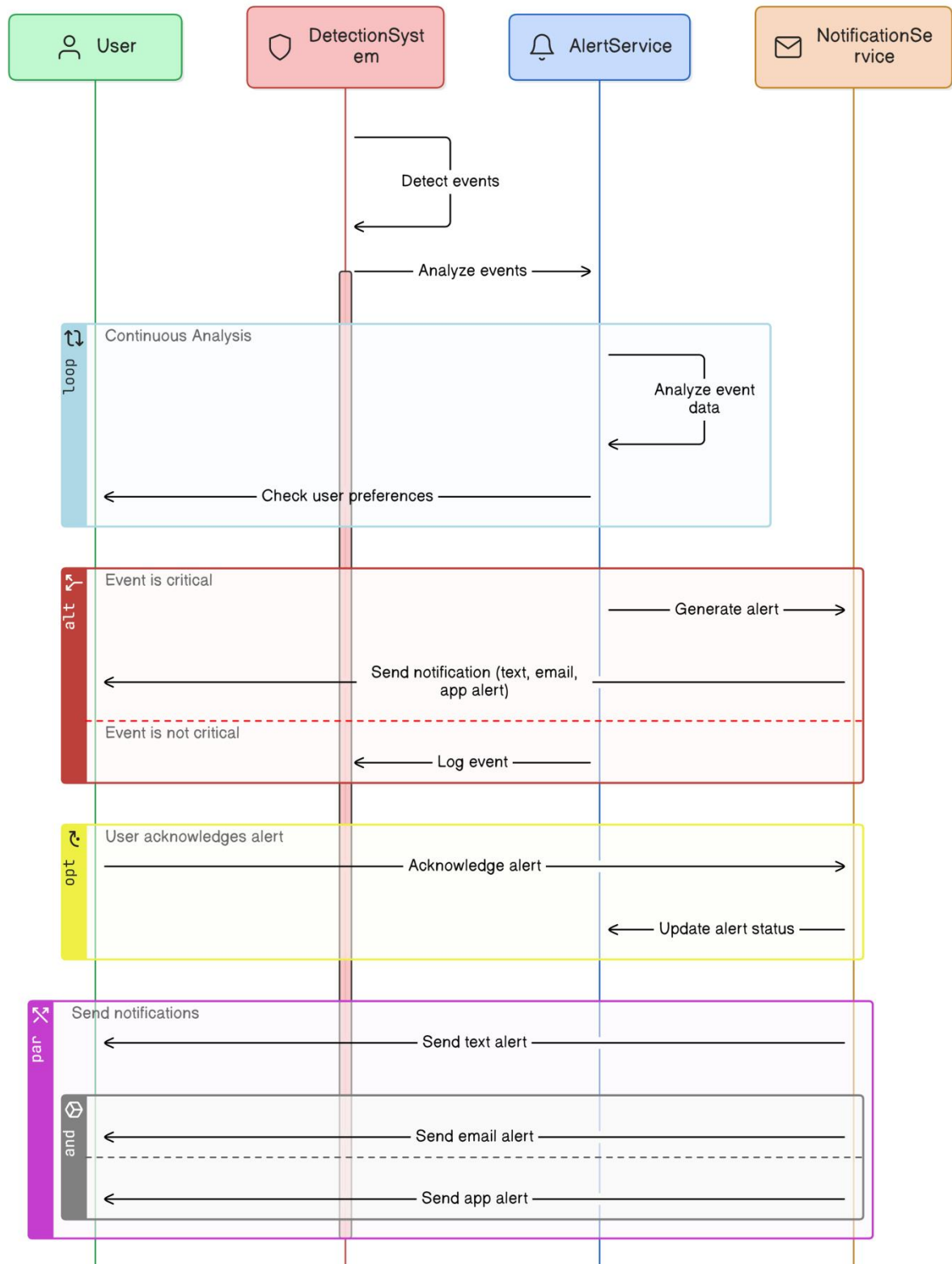


Figure 8 sequence diagrams

Dual Mode Operation

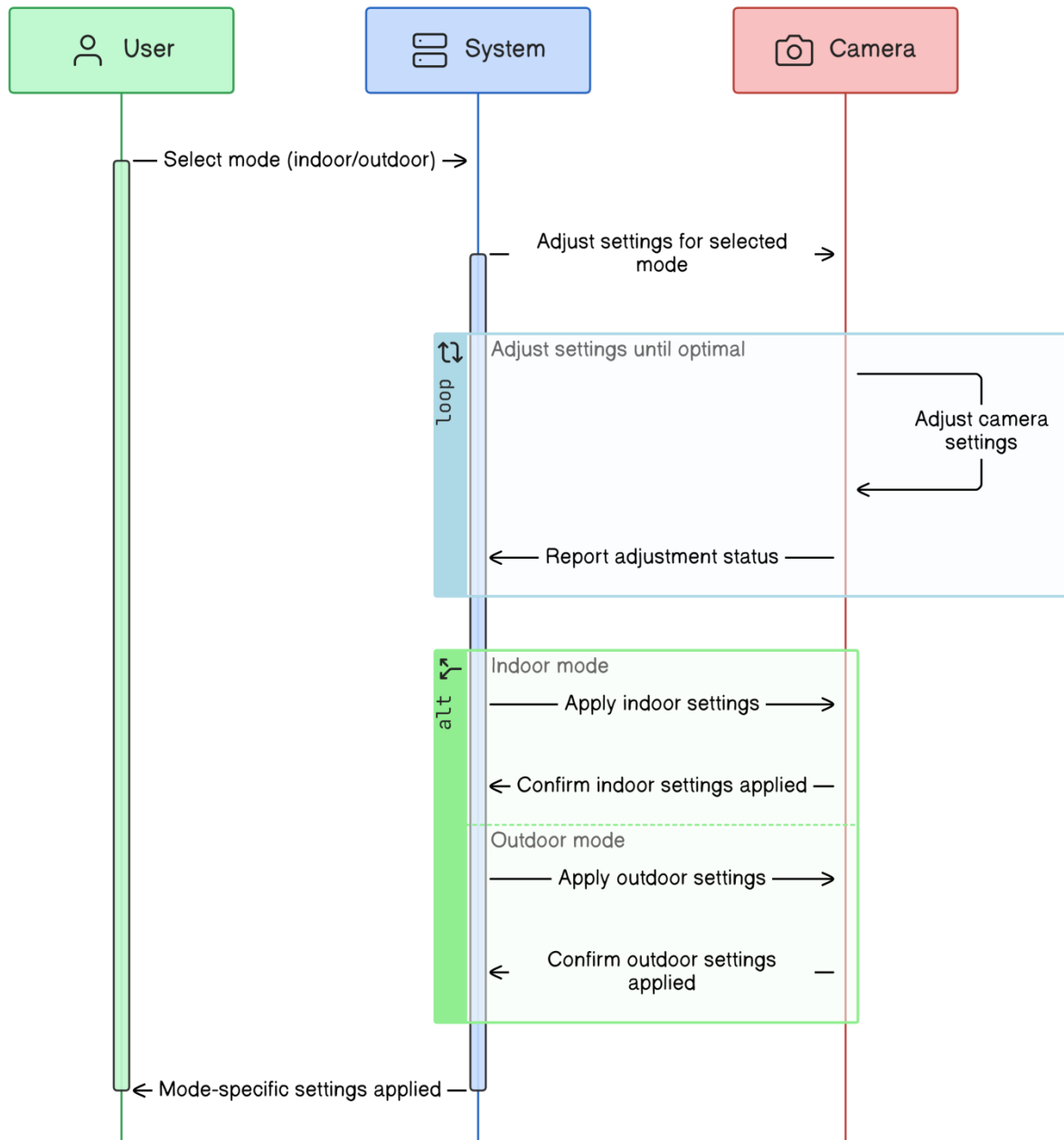


Figure 9 sequence diagrams

2.4 Tools, libraries, special algorithms and implementation environment

01. ESP32-CAM WIFI IP Camera

- ❖ Hardware: ESP32-CAM module with a 2MP camera
- ❖ Firmware: Custom firmware for video streaming and relay control.
- ❖ Libraries:
 - ESP-IDF: Espressif IoT Development Framework for developing applications for the ESP32.
 - Arduino IDE: For easy development and uploading code to the ESP32-CAM.

2. AI-Powered Object Detection

- ❖ Algorithm: YOLOv8 (You Only Look Once, version 8)
- ❖ Libraries:
 - OpenCV: For handling image processing tasks.
 - TensorFlow or PyTorch: Frameworks for implementing and running the YOLOv8 model.
 - Darknet: YOLO implementation that can be used for training and running YOLO models.
- ❖ Tools:
 - Google Colab or Jupyter Notebook: For training the YOLOv8 model.
 - LabelImg: For annotating training data if needed.

3. Cloud-Based Live Streaming

- ❖ Services:
 - AWS (Amazon Web Services): For cloud server infrastructure.
 - Firebase: For real-time database and authentication.
- ❖ Libraries:
 - FFmpeg: To handle video encoding and streaming.
 - WebRTC: For real-time communication, enabling live video streaming over the web.

4. Video Recording and Playback

❖ Libraries:

- FFmpeg: For video recording and playback functionalities.
- Local Storage: Integration with the ESP32-CAM's SD card or cloud storage for video storage.

❖ Tools:

- Node.js: For managing backend services related to video recording and playback.

5. Programmable Physical Relay Control

❖ Hardware: Relay modules compatible with the ESP32-CAM.

❖ Libraries:

- ESP32 GPIO Library: For controlling GPIO pins to manage relay modules.

❖ Tools:

- Arduino IDE: For programming relay control logic.

6. Flutter Mobile Application

❖ Framework: Flutter for cross-platform mobile app development.

❖ Libraries:

- Dart: Programming language used by Flutter.
- Provider: For state management in Flutter.
- Flutter Blue: For Bluetooth communication if needed.
- HTTP: For API calls and interacting with the backend services.

❖ Tools:

- Android Studio or Visual Studio Code: IDEs for Flutter development.

7. Security Alerts and Notifications

❖ Libraries:

- Firebase Cloud Messaging (FCM): For sending push notifications.
- Twilio: For SMS notifications.
- SendGrid: For email notifications.

❖ Tools:

- Python: For backend services that handle alert logic and notifications.

Implementation Environment

❖ Development Environment:

- Local: For ESP32-CAM development and testing.
- Cloud: AWS 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.
- **Accessing recorded footage** from the mobile app, as well as starting and stopping video recordings. These activities ensure the user can respond quickly to threats and maintain security at all times.

Contexts

The system operates in diverse environments such as homes, offices, warehouses, and public places. 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 10 interfaces



Figure 11 interfaces



Figure 12 interfaces



Figure 13 interfaces



Figure 14 interfaces



Figure 15 interfaces

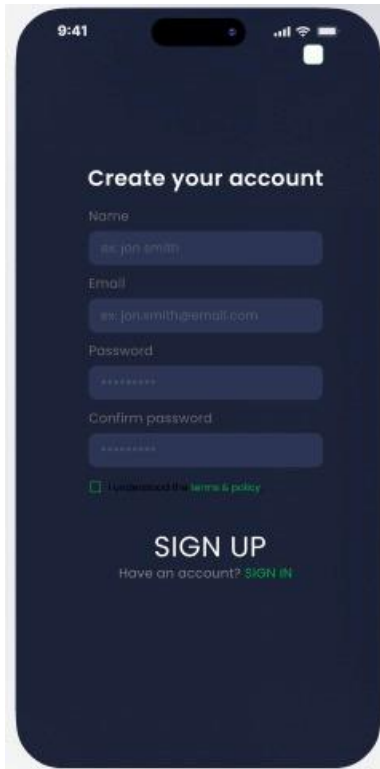


Figure 16 interfaces

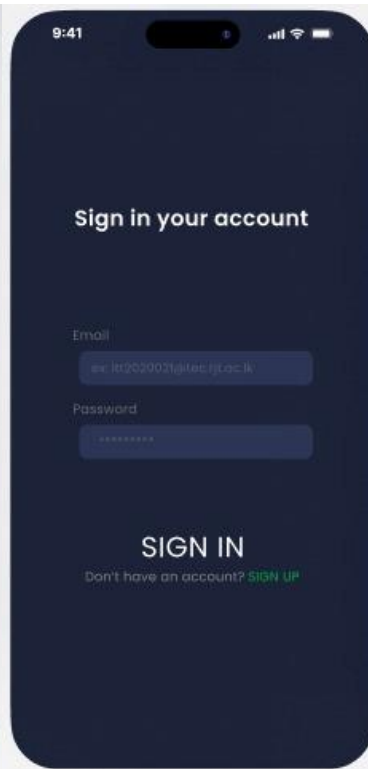


Figure 17 interfaces



Figure 18 interfaces



Figure 19 interfaces



Figure 20 interfaces

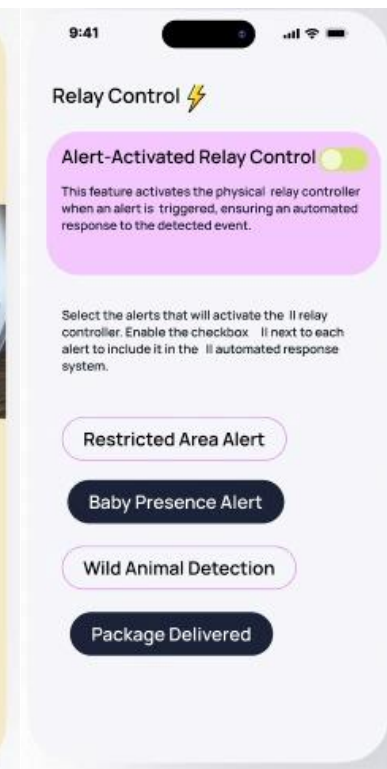


Figure 21 interfaces

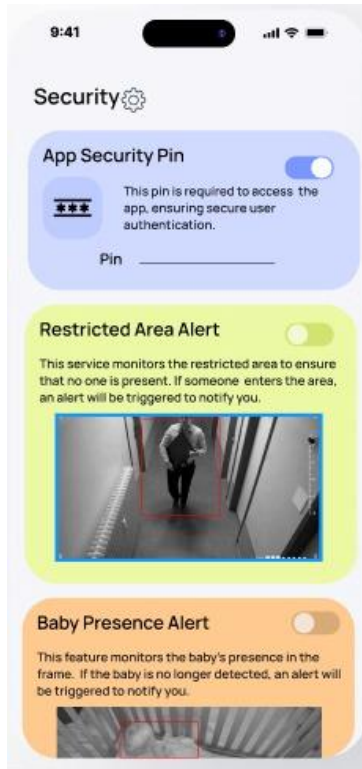


Figure 22 interfaces



Figure 23 interfaces



Figure 24 interfaces



Figure 25 interfaces



Figure 26 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

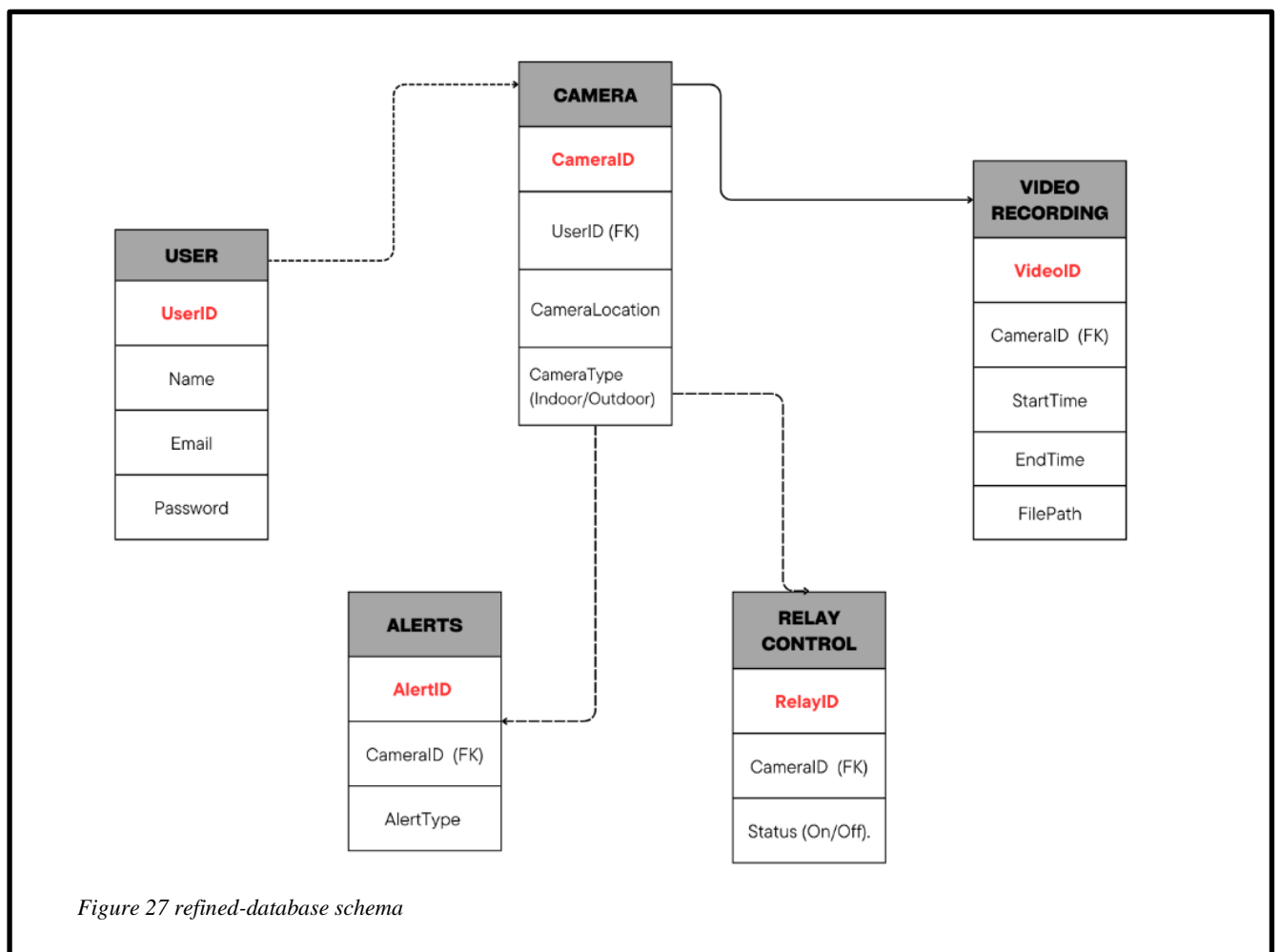
❖ Relationships

- 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



4.4 Physical database design

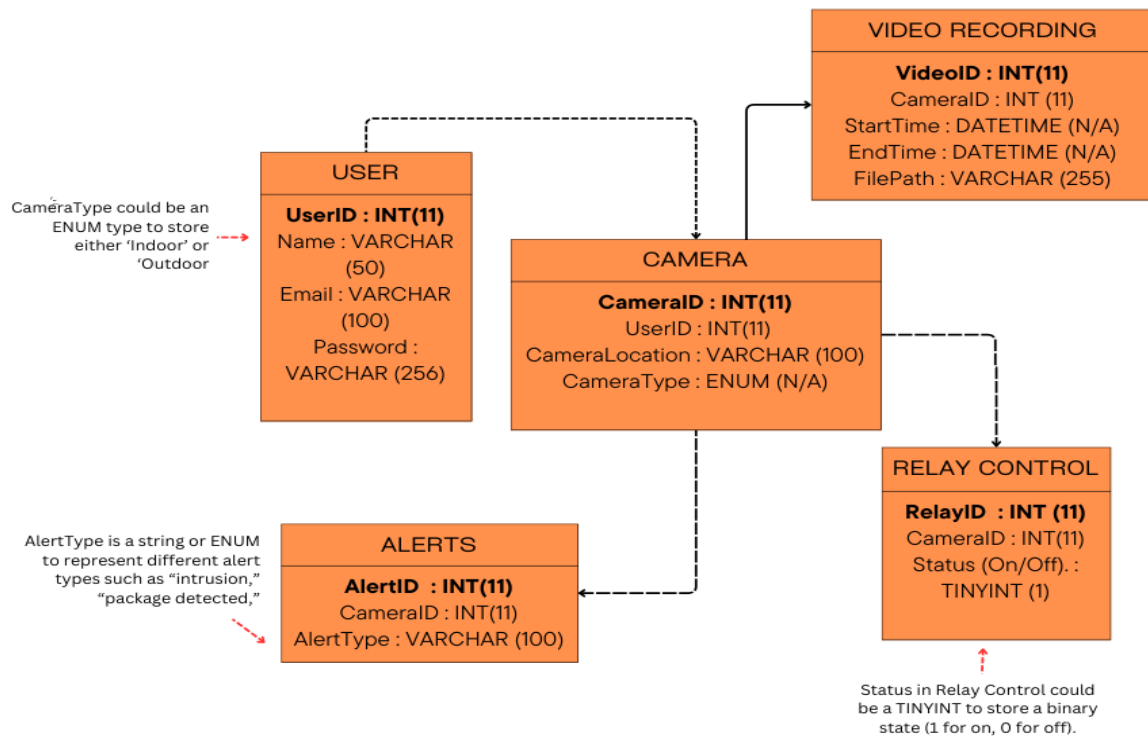


Figure 28 Physical ER diagram

5 Hardware Design

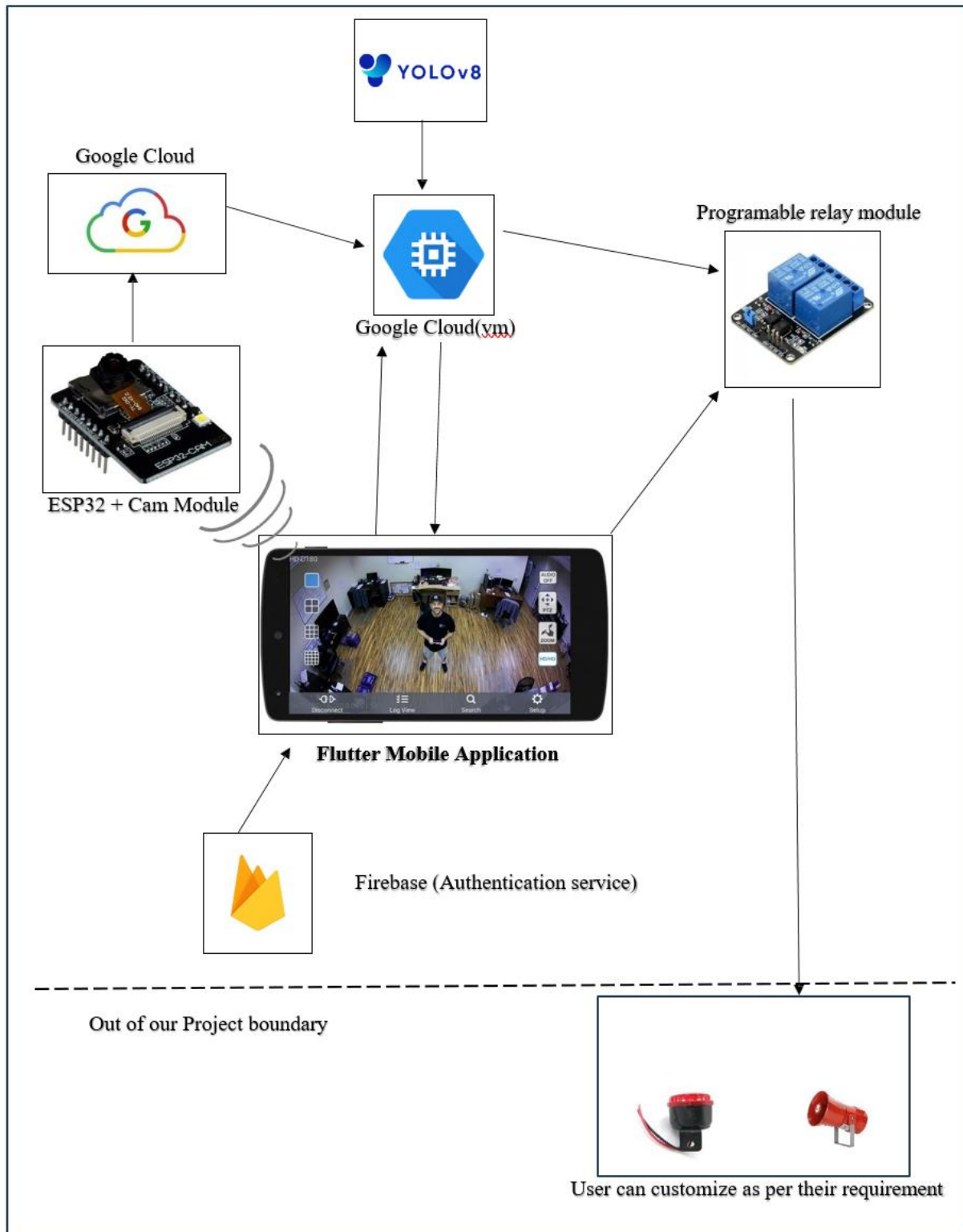


Figure 29 architecture diagrams

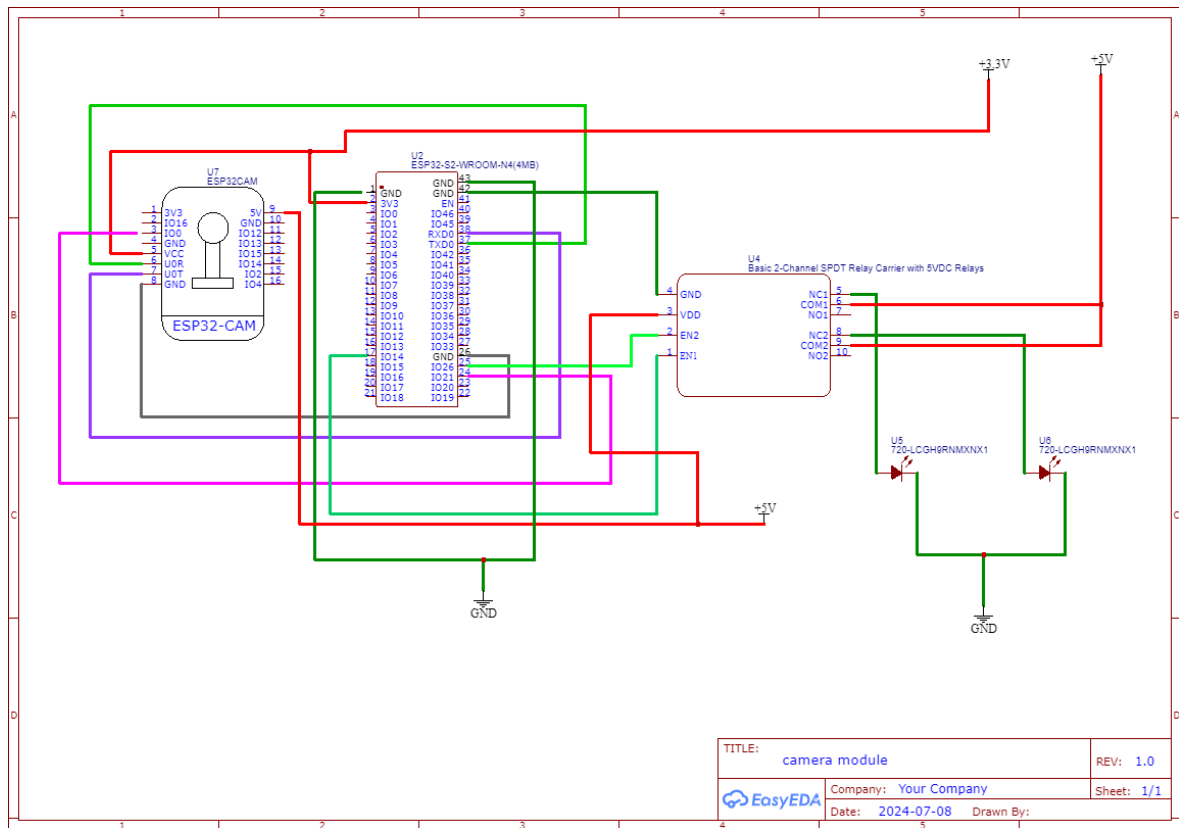


Figure 30 Hardware Design

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 | |