* **Module – Computer project**
* **Project name- Smart door delivery authentication system**
* **Team number – 2**
* **Team members**

**Banuka Abeysundera – Team leader  
Umair Mohommeth**

**Gihan Atupolage  
Senuri Minhari**

Acknowledgement

We would like to express our deepest gratitude to our project supervisor, for the invaluable guidance, feedback, and support throughout the development of this project. We are also thankful to the faculty and staff of ICBT for providing the necessary resources. Finally, we appreciate our team members' hard work and dedication in making this project a success.

Contents

[Introduction 2](#_Toc198238367)

[Problem Statement 3](#_Toc198238368)

[Objectives 3](#_Toc198238369)

[Primary Objective: 4](#_Toc198238370)

[Sub-Goals: 4](#_Toc198238371)

[Literature Review 5](#_Toc198238372)

[Proposed Solution 7](#_Toc198238373)

[Functionalities 7](#_Toc198238374)

[Functional requirements 7](#_Toc198238375)

[Non-Functional Requirements 8](#_Toc198238376)

[Software Development Methodology 8](#_Toc198238377)

[Feasibility Study Report 8](#_Toc198238378)

[1.1. Technical Feasibility 9](#_Toc198238379)

[1.2. Economic Feasibility 9](#_Toc198238380)

[1.3. Operational Feasibility 9](#_Toc198238381)

[1.4. Legal & Social Feasibility 9](#_Toc198238382)

[Requirement Gathering 9](#_Toc198238383)

[Questionnaire (20 Questions) 10](#_Toc198238384)

[Summary of Insights 13](#_Toc198238385)

[Resource Identification 14](#_Toc198238386)

[Hardware Components 14](#_Toc198238387)

[Software & Development Tools 15](#_Toc198238388)

[Human Resources (Team Roles) 15](#_Toc198238389)

[Work Breakdown Structure (WBS) 15](#_Toc198238390)

[Risk Analysis 16](#_Toc198238391)

[Design of the System 17](#_Toc198238392)

[Database Design 17](#_Toc198238393)

[Entity Relationship Diagram (ERD) 18](#_Toc198238394)

[Normalized Relational Schema 18](#_Toc198238395)

[System Design 21](#_Toc198238396)

[Data Flow Diagrams 21](#_Toc198238397)

[Context Level DFD 22](#_Toc198238398)

[LEVEL 0 DFD 23](#_Toc198238399)

[LEVEL 1 DFD 24](#_Toc198238400)

[Co-Functionalities (Non-Functional Requirements) 27](#_Toc198238401)

[Limitations of the Project 30](#_Toc198238402)

[Lessons Learned 31](#_Toc198238403)

[Future Recommendations 33](#_Toc198238404)

[Conclusion 34](#_Toc198238405)

# Introduction

In the modern era of e-commerce, doorstep delivery has become an integral part of urban life. However, with this convenience comes the risk of parcel thefts, misplacements, and unauthorized access. This project introduces the Smart Doorstep Delivery Authentication System (SDDAS), an affordable IoT-based solution that secures parcel deliveries and provides real-time alerts and digital evidence in case of tampering or unauthorized pickups. The system utilizes embedded hardware like ESP32-CAM, RFID, weight sensors, and IR modules to automate and monitor delivery and pickup phases securely.

# Problem Statement

The rapid growth of e-commerce has made doorstep delivery a major vulnerability. Unattended parcels are frequently stolen, tampered with, or picked up by the wrong person—especially in shared or urban spaces. Traditional CCTV systems offer limited help, as they passively record without preventing theft or providing timely alerts.

Key issues include:

* No identity verification during delivery or pickup
* No record of who collected the parcel or when
* Victims lack evidence or accountability for stolen packages

Most affordable systems lack real-time alerts, access control, or reliable authentication. Cybersecurity is often overlooked, making many IoT solutions unfit for secure use.

There is a clear need for a smart, low-cost system that not only monitors but also secures deliveries—through authentication, tamper detection, real-time alerts, and secure logging—to ensure safe and verifiable parcel handoffs at the doorstep.

# Objectives

The primary objective and sub-goals of the Smart Doorstep Delivery Authentication System (SDDAS) are outlined below to ensure a secure, reliable, and low-cost IoT-based delivery management system.

## Primary Objective:

To design and implement a smart, cyber-secure parcel container that uses ESP32-CAM and sensors to authenticate owners, detect tampering, and provide real-time alerts and evidence during delivery and pickup.

## Sub-Goals:

Secure Parcel Delivery

  • Enable parcel delivery detection through IR sensor and weight sensor (load cell).

  • Capture an image of the delivery person using the internal ESP32-CAM.

  • Automatically lock the container after parcel placement is detected.

Owner Authentication for Pickup

  • Allow only registered users to unlock the container using an RFID tag.

  • Record the identity confirmation before the parcel is removed.

Cyber-Tamper Detection

  • Identify unauthorized access when a parcel is removed without prior RFID scan.

  • Capture and log theft attempts with time-stamped images and event data.

Dual-Camera Evidence Capture

  • Use ESP32-CAM to capture images of the person.

  • Use a secondary camera to monitor the parcel from above or side-view during delivery and pickup.

Real-Time Alerts and SD Card Logging

  • Send images and status updates to the owner via Telegram bot for all events.

  • Log all events with corresponding images and timestamps to SD card for offline access and audit.

Cost Efficiency and Offline Functionality

  • Ensure the use of low-cost components without compromising functionality.

  • Maintain basic local logging and security features even if internet connectivity is lost.

# Literature Review

With the rise of e-commerce and contactless delivery, the need for secure, unattended parcel handling is more relevant than ever. While several technologies address parts of this challenge, few offer a complete solution combining physical authentication, tamper detection, and secure communication.

Smart Lockers

* Singh and Rani (2020) developed an IoT-based smart locker using RFID and OTP access via a centralized app. However, it lacked tamper detection and offline fail-safes.
* Reference: Singh, A. and Rani, R., 2020. IoT based Smart Locker System for Secure Parcel Delivery. International Journal of Engineering Research & Technology (IJERT), 9(4).

Surveillance Systems

* Chen, Li, and Zhang (2019) reviewed intelligent video systems, highlighting their enhanced detection but also noting high cost, power use, and privacy issues.
* Reference: Chen, Y., Li, X. and Zhang, H., 2019. Intelligent Surveillance Systems: Challenges and Trends. Journal of Sensors, Hindawi.

Smart Drop Boxes

* Gordon et al. (2021) presented a motion-sensor-based smart drop box. It offered alerts but lacked recipient authentication and platform integration.
* Reference: Gordon, D., Myers, L. and Shah, T., 2021. Design of a Smart Parcel DropBox Using IoT. In: 2021 IEEE SmartTech Conference.

IoT Cybersecurity

* Alrawais et al. (2017) stressed the importance of local data logging and secure access control in IoT systems, aligning with this project's SD card and RFID use.
* Reference: Alrawais, A., Alhothaily, A., Hu, C. and Cheng, X., 2017. Fog Computing for the Internet of Things: Security and Privacy Issues. IEEE Internet Computing, 21(2), pp.34–42.

Telegram Alerts

* Kumar and Srinivasan (2022) demonstrated that Telegram bots provide secure, fast IoT messaging—superior to email or SMS.
* Reference: Kumar, V. and Srinivasan, R., 2022. IoT-Based Alert System using Telegram Bot. International Journal of Computer Applications, 184(17).

Multi-Factor Authentication

* Pradeep and Khan (2023) found RFID and weight sensing effective for user validation in IoT systems, supporting multi-sensor approaches.
* Reference: Pradeep, M. and Khan, R., 2023. Smart Inventory Tracking with Multi-Factor Authentication using IoT. International Journal of Emerging Trends in Engineering Research, 11(2).

Conclusion:

Most existing systems focus on isolated features. This project uniquely combines multi-factor authentication, dual-camera monitoring, SD-based logging, and Telegram alerts into a single, cost-effective smart delivery solution.

# Proposed Solution

We propose a complete embedded system that automates parcel drop detection, authenticates pickup via RFID, captures and logs images of all actors (delivery person, recipient, or intruder), and sends them to the owner via Telegram. The system works entirely on local storage for reliability and uses Wi-Fi only for push notifications. Dual-camera setup ensures both human and parcel spaces are captured. SD card storage ensures forensic logs even without Wi-Fi.

# Functionalities

## Functional requirements

* Detect motion using IR sensor
* Capture person photo via Cam 1 (delivery, pickup, theft)
* Detect parcel placement/removal via weight sensor
* Capture parcel image via Cam 2
* RFID authentication before pickup
* Trigger Telegram alert with photo evidence
* Store images and event logs on SD card
* Lock/unlock container with servo based on state transitions

## Non-Functional Requirements

* System should operate offline with SD logging if Wi-Fi is down
* Telegram alerts must be encrypted and private
* Must function within 1–2 seconds per action
* Entire system must operate on a low-power microcontroller
* Deployment cost should remain below budget
* The system should be easy to install and weather-resistant

# Software Development Methodology

The Incremental Software Development Model has been selected for the development of the Smart Doorstep Delivery Authentication System (SDDAS). This model structures the system into discrete, functional components that are designed, implemented, and tested in successive increments. Each increment delivers a working subset of the final product—such as delivery detection, parcel logging, pickup authentication, and theft detection—allowing for early validation, continuous refinement, and smooth integration of features. The model is particularly well-suited for embedded IoT projects like SDDAS, where hardware and software must be developed in parallel with frequent testing. By enabling modular development, risk isolation, and efficient task division among team members, the incremental approach ensures both system reliability and development agility.

# Feasibility Study Report

* 1. Technical Feasibility  
     The system is built using ESP32-CAM modules with integrated Wi-Fi, support for SD card logging, and GPIO pins for peripheral interfacing (RFID, IR sensor, load cell, etc.). The project uses lightweight open-source libraries (ESP32 Arduino core, RFID MFRC522 library, HX711 for weight sensing, etc.), allowing seamless integration and low-level hardware control through Arduino IDE. This makes the development process feasible using basic embedded knowledge and supports incremental testing and deployment.
  2. Economic Feasibility  
     All components used in the project are low-cost and widely available. The total estimated cost for hardware is under LKR.9300 , meeting the cost-efficiency requirement. No cloud services or paid APIs are used. Telegram is free for messaging and bot integration. Storage is local (SD card), eliminating recurring hosting or server costs.
  3. Operational Feasibility  
     The system requires minimal user interaction. Delivery personnel do not need to scan anything—parcel delivery is triggered automatically by motion and weight detection. Pickup is secured with RFID, requiring only a card scan. The Telegram bot provides real-time event updates (delivery, pickup, or theft) to the owner. The system is suitable for apartments, homes, and offices with standard Wi-Fi coverage and power.
  4. Legal & Social Feasibility  
     The project ensures privacy and legal compliance by keeping all captured data local (SD card) and securely sending notifications directly to the verified owner's Telegram account. No personal data is stored in the cloud. Since the surveillance occurs on private property, it complies with local laws in most jurisdictions.

# Requirement Gathering

Method Used: Questionnaire

To understand real-world expectations and validate system requirements for the SDDAS, a questionnaire method was used. The questionnaire was shared with a focused group of 15 respondents using google forms, including frequent online shoppers, homeowners, and occasional delivery agents. Each question was designed to extract user preferences, security concerns, and expectations regarding parcel safety, notifications, and system usability.

Below are the 20 questions used, their purpose, type of responses, and summarized results:

## Questionnaire (20 Questions)

1. Have you ever had a package stolen or tampered with after it was delivered?

* Purpose: To determine how common parcel theft is among users.
* Type: Yes/No
* Result: Yes – 11; No – 5
* Insight: 67% have experienced theft, confirming the need for security features.

1. How concerned are you about the safety of your online deliveries when you’re not home?

* Purpose: To evaluate user concern for unattended deliveries.
* Type: Likert scale (1–Not concerned to 5–Extremely concerned)
* Result: Average = 3.75
* Insight: Strong concern supports the system’s relevance.

1. Would you prefer a secure container that automatically locks after a parcel is placed?

* Purpose: To gauge interest in automated physical security.
* Type: Yes/No
* Result: Yes – 12; No – 4

1. Do you support using a camera to take pictures of the delivery person during drop-off?

* Purpose: To verify user acceptance of camera surveillance.
* Type: Yes/No
* Result: Yes – 12; No – 4

1. Would you like a second camera to monitor and record the parcel itself?

* Purpose: To check interest in dual-camera monitoring.
* Type: Yes/No
* Result: Yes – 11; No – 5

1. Should the parcel container unlock only after a valid RFID scan during pickup?

* Purpose: To validate secure authentication at pickup.
* Type: Yes/No
* Result: Yes – 12; No – 4

1. How important is it to receive a Telegram alert for every delivery or pickup?

* Purpose: To evaluate the need for real-time alerts.
* Type: Likert scale (1–Not important to 5–Very important)
* Result: Average = 3.88

1. Would you prefer to receive parcel theft alerts instantly with photo evidence?

* Purpose: To determine preferences for threat notifications.
* Type: Yes/No
* Result: Yes – 13; No – 3

1. Is storing images and logs locally on an SD card acceptable for you?

* Purpose: To understand user preference for data storage.
* Type: Yes/No
* Result: Yes – 13; No – 3

1. Would you trust a Telegram bot to notify you of activity at your doorstep?

* Purpose: To measure trust in bot-based notifications.
* Type: Yes/No
* Result: Yes – 12; No – 4

1. Are you comfortable using RFID cards for authentication?

* Purpose: To ensure RFID usability.
* Type: Yes/No
* Result: Yes – 13; No – 3

1. If RFID fails, would you want an optional backup method (e.g., OTP or manual override)?

* Purpose: To plan contingency authentication.
* Type: Yes/No
* Result: Yes – 11; No – 5

1. Would you feel safer if motion was detected before unlocking the container?

* Purpose: To confirm value of IR sensor usage.
* Type: Yes/No
* Result: Yes – 12; No – 4

1. Is it useful for you to view delivery history including date, time, and pictures?

* Purpose: To understand logging importance.
* Type: Yes/No
* Result: Yes – 13; No – 3

1. Should theft alerts include both person and parcel images?

* Purpose: To verify detail level of threat logging.
* Type: Yes/No
* Result: Yes – 12; No – 4

1. Would you still use the system if it worked offline but synced later?

* Purpose: To assess need for offline capability.
* Type: Yes/No
* Result: Yes – 12; No – 4

1. Do you expect the device to be tamper-resistant or alert you during tampering?

* Purpose: To define physical security expectations.
* Type: Yes/No
* Result: Yes – 13; No – 3

1. Would you find the system more reliable if it used multiple cameras or sensors?

* Purpose: To justify multi-sensor integration.
* Type: Yes/No
* Result: Yes – 4; No – 3

1. Would you want to receive only essential notifications or full logs regularly?

* Purpose: To evaluate notification preferences.
* Type: Multiple choice (A. Essentials only, B. Full logs, C. Both)
* Result: A – 6; B – 3; C – 7

1. How much would you expect to spend on a secure delivery system like this?

* Purpose: To assess acceptable cost range.
* Type: Multiple choice (A. <$20, B. $20–$40, C. $40–$60, D. >$60)
* Result: A – 3; B – 8; C – 4; D – 1

## Summary of Insights

* Parcel Theft Is Common:  
    - 69% (11 out of 16) reported experiencing theft or tampering, showing strong demand for a secure system.
* Security Features Are in High Demand:  
    - 75% (12/16) want auto-locking containers;  
    - 75% support delivery person surveillance via camera;  
    - 69% support dual-camera systems to monitor parcels.
* RFID Authentication Is Essential:  
    - 75% (12/16) want RFID-based unlocking;  
    - 81% (13/16) are comfortable using RFID cards.
* Real-Time Alerts Are Crucial:  
    - Telegram alerts rated 3.88/5 on importance;  
    - 81% (13/16) want instant theft alerts with photos;  
    - 75% trust Telegram bots for notifications.
* SD Card Logging Is Acceptable:  
    - 81% (13/16) are comfortable with local image and log storage.
* Motion Detection Enhances Security:  
    - 75% (12/16) feel safer with motion-triggered unlocking and alerts.
* Backup and Redundancy Are Valued:  
    - 69% (11/16) want optional backup methods in case RFID fails.
* Detailed Activity History Is Important:  
    - 81% (13/16) value delivery logs with date, time, and images.
* Offline Operation Is Acceptable:  
    - 75% (12/16) are okay with delayed syncing for offline use.
* Multi-Sensor Integration Adds Reliability:  
    - Though fewer respondents (4/7) supported multi-sensor use, it still indicates niche trust in layered detection.
* Notification Preferences Are Mixed:  
    - 38% want essential alerts only,  
    - 44% prefer both essentials and full logs,  
    - 19% want full logs only.
* Cost Expectations Are Modest:  
    - 69% (11/16) expect the system to cost below $40, emphasizing the need for affordability.

Google form link – https://docs.google.com/forms/d/1Fp\_l8zFDrotrrV21\_ZaZVZpjwE1SAB9gSmPFZ7JpTAo/edit

# Resource Identification

## Hardware Components

| **Component** | **Quantity** | **Purpose** |  |
| --- | --- | --- | --- |
| ESP32-CAM Module | 1 | Main microcontroller with built-in camera |  |
| External Camera (OV2640) | 1 | Second wireless camera to monitor parcel condition |  |
| RFID Module (RC522/NFC) | 1 | RFID authentication for parcel owner |  |
| RFID Tags | 2 | For authorized users (owners) |  |
| IR Sensor Module | 1 | Detects motion for triggering camera and unlocking logic |  |
| Load Cell + HX711 Module | 1 | Detects parcel placement and removal based on weight |  |
| Micro SD Card (8GB+) | 1 | Local storage of logs and images |  |
| Micro SD Card Module | 1 | Interface between SD card and ESP32-CAM |  |
| Servo Motor (SG90) | 1 | Locks/unlocks the container lid |  |
| Power Supply (5V Adapter / Battery Pack) | 1 | Powers the system reliably |  |
| Jumper Wires & Connectors | N/A | For wiring all modules together |  |
| Parcel Locking Container | 1 | Physical box to store parcel and mount hardware |  |

## Software & Development Tools

* Arduino IDE (Open Source)
* Telegram Bot API (Free)
* Git/GitHub for version control
* Excel / Google Sheets (for log review and graphing)
* Fritzing or Proteus (for schematic diagrams)

## Human Resources (Team Roles)

| **Role** | **Responsibility** |
| --- | --- |
| Hardware Engineer | Integrate and test IR, RFID, servo, load cell |
| Embedded Developer | Write firmware logic for ESP32 |
| Software Developer | Telegram bot, logging, SD card handling |
| QA & Documentation Analyst | Test all phases, write documentation, analyze risks |

# Work Breakdown Structure (WBS)

| **Task ID** | **Description** | **Assigned Role** | **Deliverable** |
| --- | --- | --- | --- |
| T1 | Define requirements via questionnaire | QA/Analyst | Requirement Report |
| T2 | Set up hardware circuits | Hardware Engineer | Functional prototype |
| T3 | Integrate IR, Cam 1, servo (Delivery phase) | Embedded Dev | Delivery phase complete |
| T4 | Integrate Cam 2, RFID, load cell (Pickup phase) | Embedded Dev | Pickup phase complete |
| T5 | Develop SD logging & Telegram alerts | Software Dev | Complete alert & log system |
| T6 | Final system test & failure simulation | QA | Bug Report, Test Cases |
| T7 | Document design & submit reports | All team | Final Report, Diagrams |

# Risk Analysis

| **Risk** | **Likelihood** | **Severity** | **Mitigation** |
| --- | --- | --- | --- |
| IR sensor false positives | Medium | Moderate | Add timer/debounce logic |
| RFID reader failure | Low | High | Add retry logic, secondary verification |
| Telegram delay or no connection | Medium | Medium | SD card backup always active |
| Weight sensor drift or error | Low | Moderate | Calibration and margin thresholding |
| Image file corruption on SD card | Low | High | Save checksum logs and retry writes |
| Power outage during process | Medium | High | Option to integrate small UPS or rechargeable battery module |

# Design of the System

## Database Design

Since this is an embedded system using SD card storage (no cloud database), the system stores structured log entries as text files (or CSV) and associated images (JPEG) on the SD card. Each log entry contains event metadata and a file reference to the captured image.

### Entity Relationship Diagram (ERD)

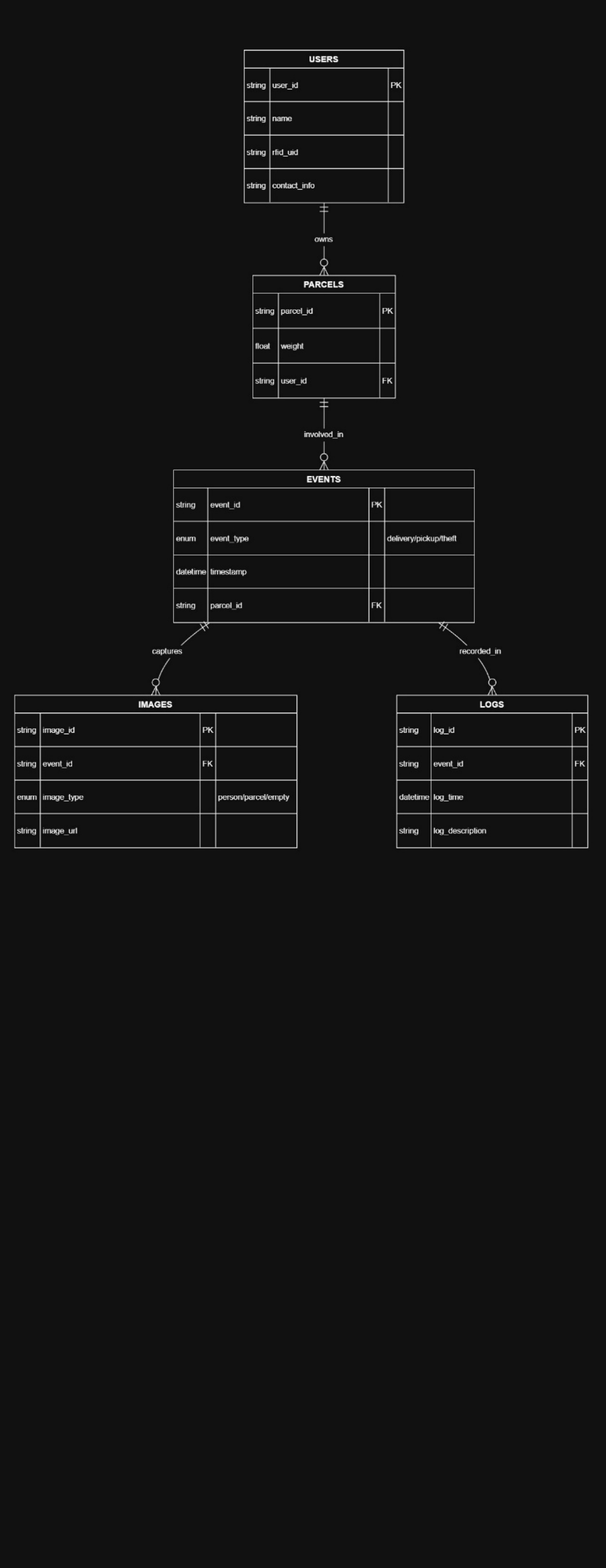


Figure -ER Diagram

### Normalized Relational Schema

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |



First Normal Form (1NF)

All values are atomic. No repeating groups or multivalued attributes.

Second Normal Form (2NF)

All non-key attributes are fully functionally dependent on the whole primary key.

No changes from 1NF, since all tables use simple (non-composite) primary keys.

(Structure remains same as 1 NF)

Third Normal Form (3NF)

No transitive dependencies. All non-key attributes are only dependent on the primary key.

Already satisfied in 2NF. No schema changes needed.

* USERS: all fields depend on userjd
* PARCELS: all fields depend on parceljd
* EVENTS: all fields depend on eventjd
* IMAGES: all fields depend on imagejd
* LOGS: all fields depend on logjd

Final structure is fully normalized to 3NF.

# System Design

We will represent the system using both DFD and UML. This combination gives both the process perspective and the object/interaction view.

## Data Flow Diagrams

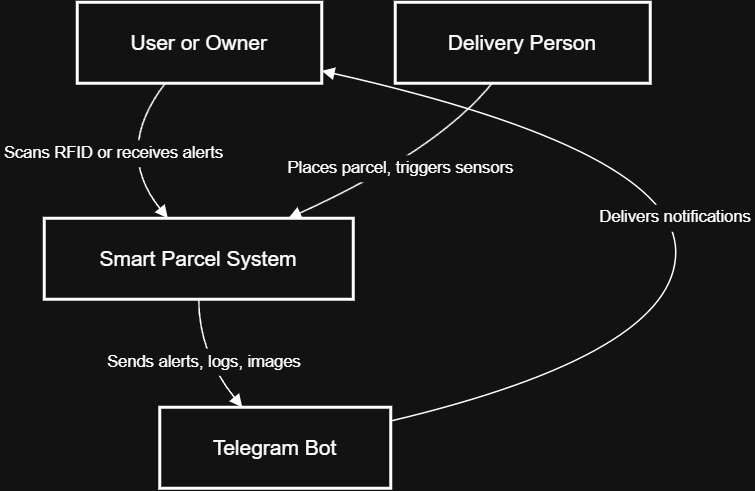
Context Level DFD

Figure - Context Level Diagram

Data Flow Summary:

* Delivery triggers IR Sensor → Cam 1 takes picture → Parcel placed → Cam 2 captures → Log saved to SD + Telegram Alert
* Pickup IR Trigger → Cam 1 captures → RFID scanned → If valid, box unlocks → Parcel removed → Cam 2 captures → Log saved → Telegram Alert

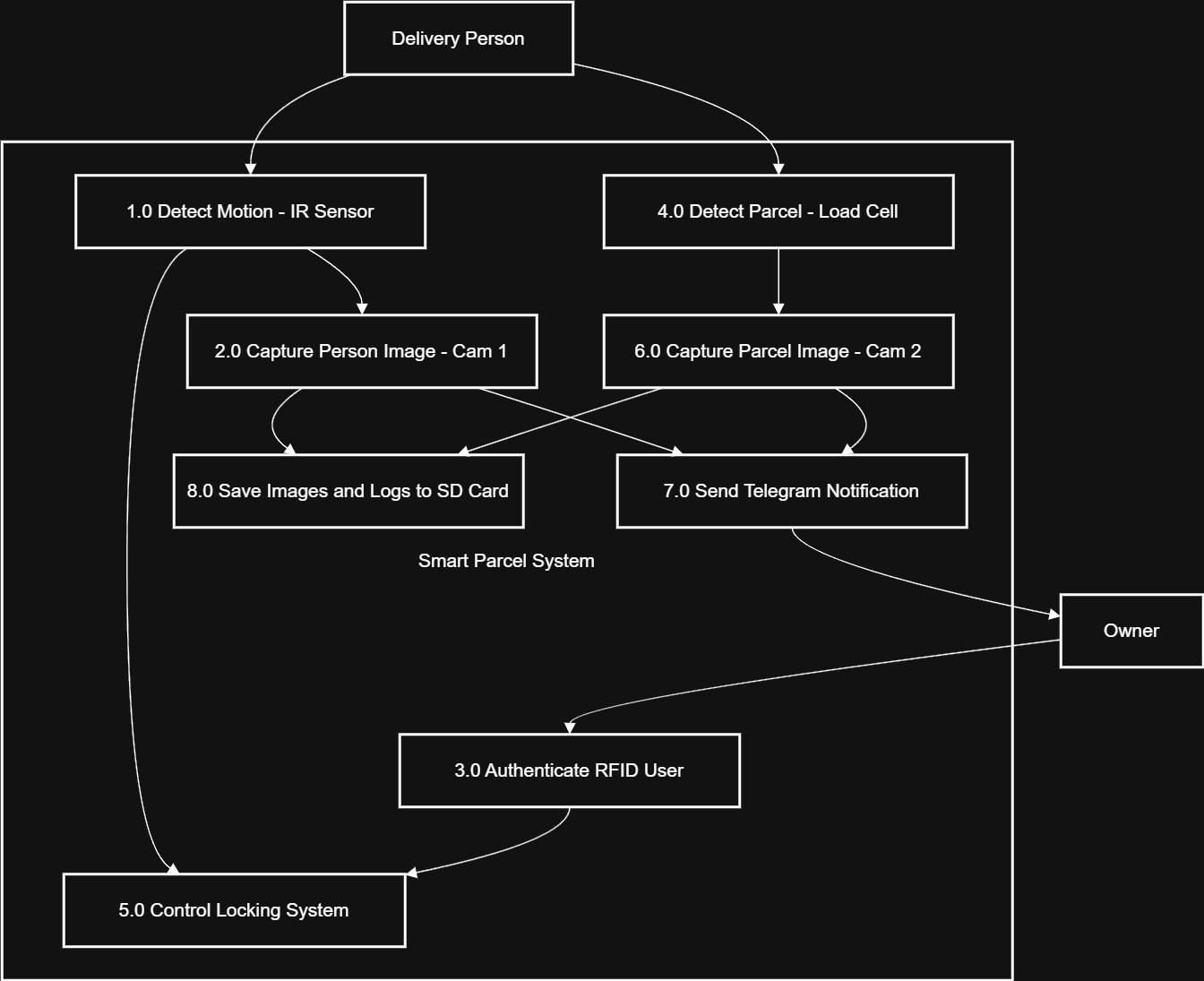
LEVEL 0 DFD

Figure - Level 0 Diagram

### LEVEL 1 DFD

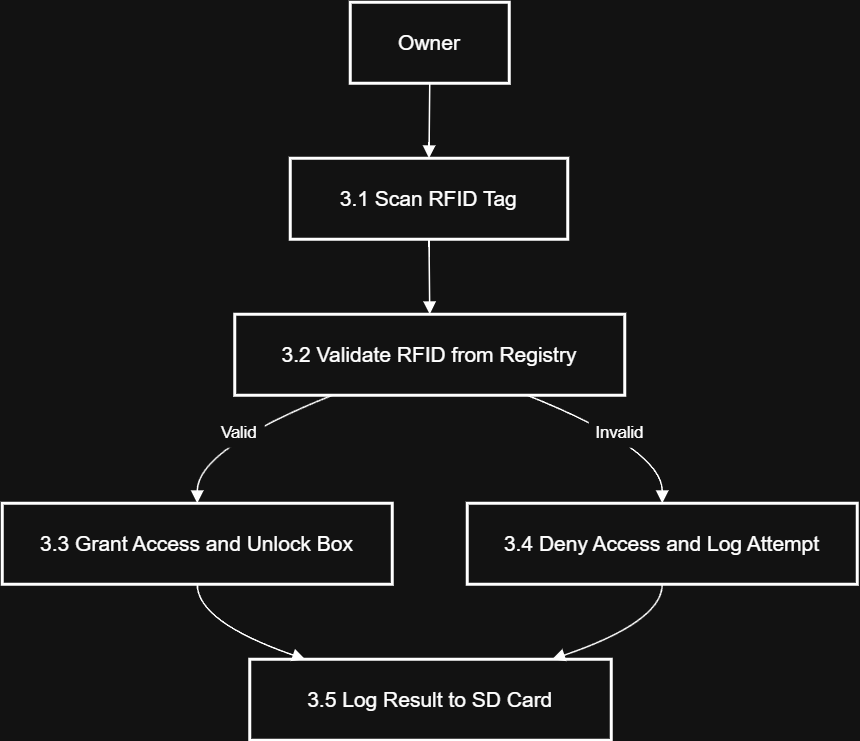


Figure - LEVEL 01

UML Diagrams

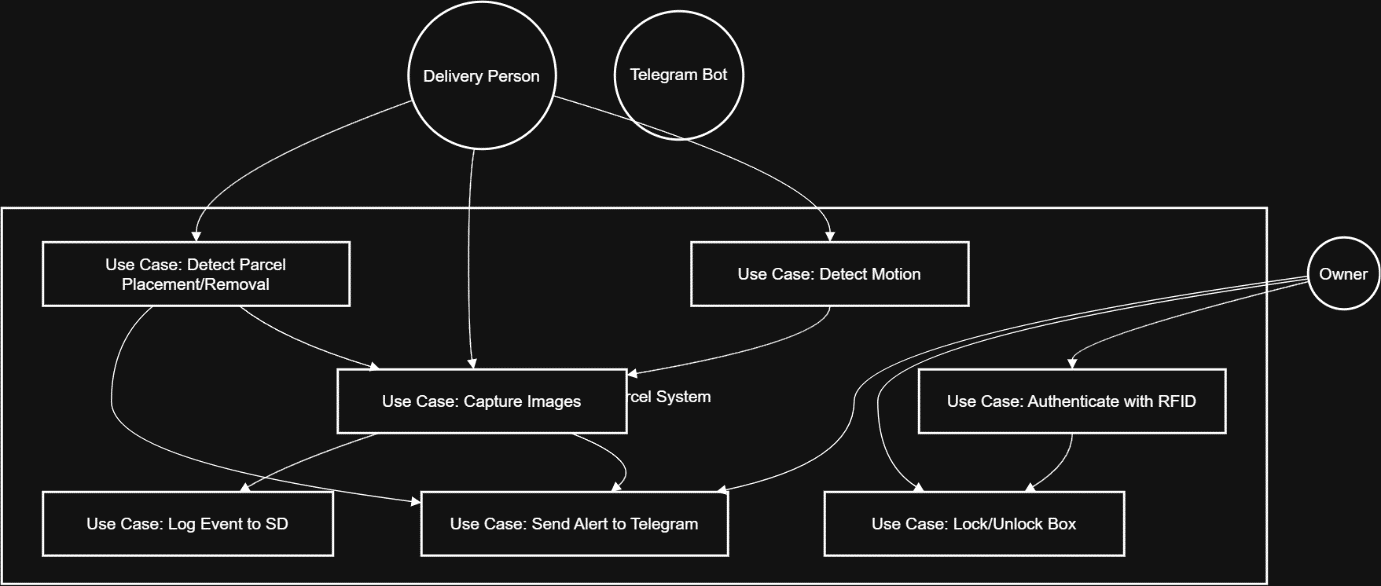


Figure - Case Diagram

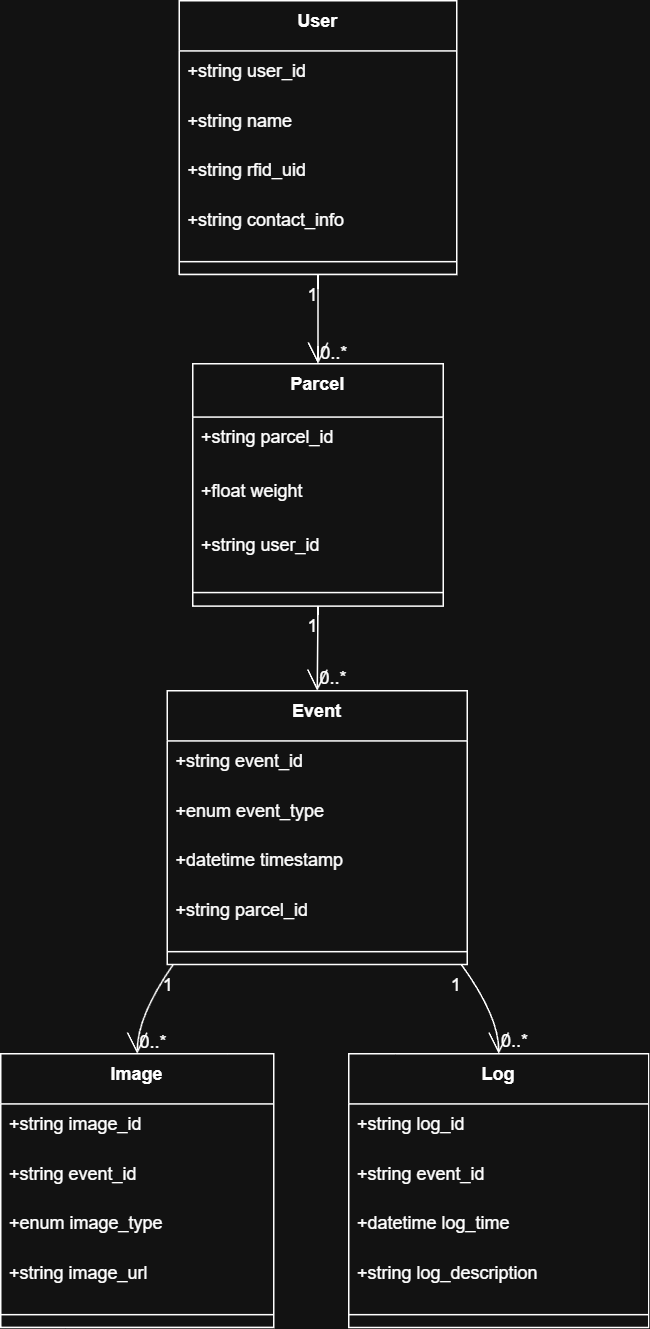


Figure - Class diagram

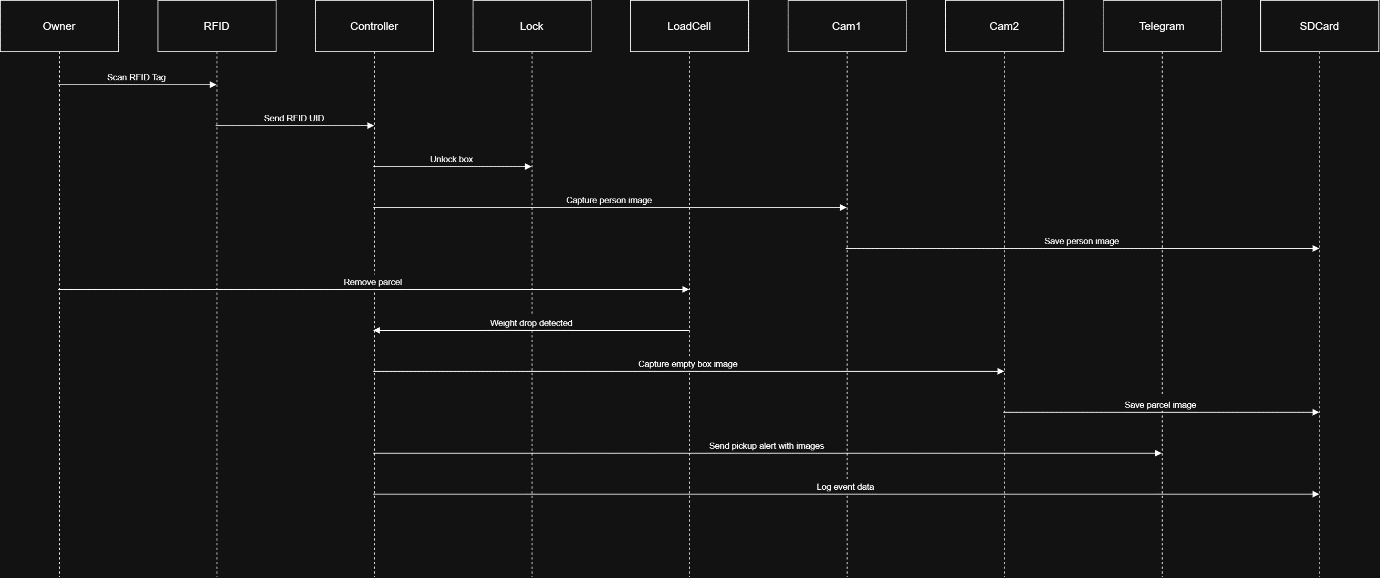


Figure - Sequence diagram

# Co-Functionalities (Non-Functional Requirements)

The following co-functionalities have been identified and designed to support the core functionality of the Smart Doorstep Delivery Authentication System (SDDAS). These features improve reliability, usability, and cybersecurity compliance. Since the project is still under development, the following points are based on planned design behavior, early component testing, and references to similar systems.

Reliability

* Expected Behaviour: The system will consistently detect parcel placement/removal, trigger alerts, and activate servo locking.
* Planned Verification: Individual modules (IR sensor, load cell, RFID) are being tested for stability under repeated use.
* Justification: Ensures that the system performs expected operations without failure.

Real-Time Alerting

* Expected Behaviour: The system will send alerts to the parcel owner via Telegram within 5 seconds of an event (delivery, pickup, or theft).
* Planned Verification: Telegram Bot API will be integrated with ESP32-CAM for testing message/photo delivery speed.
* Justification: Timely alerts allow the owner to monitor the parcel in real time and act if needed.

Security & Authentication

* Expected Behaviour: Only the parcel owner with a valid RFID tag will be able to unlock the container to retrieve the parcel.
* Planned Verification: RFID module has been tested to detect UID and trigger output. Full integration with the servo lock is scheduled for the next phase.
* Justification: Prevents unauthorized access and adds a layer of cybersecurity through local authentication.

Data Logging and Backup

* Expected Behaviour: Every event (delivery, pickup, unauthorized access) will be logged to an SD card with timestamps and associated images.
* Planned Verification: SD card module will be configured with FAT32 formatting to write log.txt and image files.
* Justification: Ensures data is available for future review even if internet or Telegram fails.

Usability

* Expected Behaviour: Minimal user interaction — delivery person only places the parcel; parcel owner scans RFID to unlock.
* Planned Observation: During pilot testing, ease-of-use will be measured by asking users to complete pickup/delivery actions without instruction.
* Justification: Encourages practical use in real households without requiring technical skills.

Low Power Operation

* Expected Behaviour: System operates on low power (via adapter or power bank) for 24/7 monitoring.
* Assumption: ESP32-CAM consumes ~180–250mA during Wi-Fi operations; confirmed through data sheets.
* Justification: Ensures uninterrupted operation and off-grid compatibility with a large power bank or solar.

Cost Efficiency

* Design Constraint: All components chosen for affordability and availability.
* Current Cost Estimate: Under LKR.9300 total for ESP32-CAM, IR sensor, RFID module, servo, load cell, HX711, and SD card module.
* Justification: Makes the solution accessible for widespread adoption by individuals and small businesses.

Privacy Control

* Expected Behaviour: No cloud services (like Firebase) used; only Telegram and local SD storage are used for image and log handling.
* Justification: Reduces risk of external data leaks and gives the owner full control over the system.

Expandability

* Expected Behaviour: System will allow adding modules (e.g., buzzer, GSM, keypad) via GPIO or I2C.
* Design Note: Unused GPIOs have been reserved for future expansions.
* Justification: Supports customization and future enhancements.

Maintainability

* Design Approach: Modular code and hardware layout. Each sensor/module can be tested or replaced independently.
* Planned Implementation: All functional blocks (RFID, IR, Load Cell, Camera, Notification) will have separate functions and headers in code.
* Justification: Simplifies future debugging, updates, and maintenance.

# Limitations of the Project

Despite the successful implementation of the SDDAS system and its tested functionality, the project does have a few practical limitations due to hardware, cost, and scalability constraints:

Hardware Dependency:

The project is highly dependent on components like the ESP32-CAM and RFID scanner. A single point of failure (e.g., camera or sensor) can compromise the security.

Limited Parallel Access:

The system can only handle one delivery or pickup event at a time. If two people approach simultaneously, only the first interaction is processed accurately.

Camera Resolution and Angle:

The ESP32-CAM has limited camera quality, which may affect image clarity in low lighting or from odd angles. This could hinder facial or object recognition in the future.

Storage Limitation:

While logs are saved on the SD card, frequent image capture may fill the card quickly. There is no auto-overwrite or archive mechanism currently.

Telegram Reliance:

If internet access is unavailable or the Telegram bot is inactive, the user will not receive real-time alerts. Although logs are stored offline, immediate action is compromised.

Power Supply:

The current system requires a stable DC power source. Battery operation is possible but may lead to quicker depletion due to camera and Wi-Fi usage.

RFID Security:

RFID tags/cards, if lost or cloned, can be used to access the parcel box unless replaced or blocked manually by the user.

# Lessons Learned

Through the implementation of this project, several technical and project management insights were gained:

Real-World Hardware Synchronization:

Managing real-time synchronization between hardware components (IR, RFID, Cameras, Load Cell) requires careful timing and delay handling in code.

Value of Modular Development:

Following the Incremental SDLC model helped isolate and test modules independently before integrating, leading to fewer critical bugs.

Importance of Redundancy:

Incorporating both Telegram alerting and SD card logging ensured event traceability even in offline conditions.

User Feedback:

Initial user testing through questionnaires revealed that parcel theft and missed deliveries are real-world issues, validating the system's necessity.

Environmental Factors:

Light, temperature, and angle significantly affect the performance of cameras and IR sensors, which must be considered in future designs.

Documentation and Planning:

Clear task division among team members ensured timely development and thorough documentation.

# Future Recommendations

To improve the system and expand its utility, the following enhancements are recommended:

AI-Based Facial Recognition:

Integrate facial recognition to verify the identity of delivery personnel and authorized users, increasing security beyond RFID.

Cloud Backup & Dashboard:

Sync logs to a free cloud service like Google Drive or Firebase (optional tier) for remote historical access.

Multi-Parcel Handling:

Enhance the box to handle multiple parcels and map weight-to-owner through a QR or barcode system (if future support for delivery-side authentication is possible).

Solar Power Integration:

Make the system eco-friendly and off-grid capable with solar charging and a battery pack.

SMS/Email Notification:

Add additional notification channels besides Telegram to ensure delivery visibility.

Camera Night Vision:

Add IR-based night vision modules or low-light sensors to improve image quality in poor lighting.

Admin Panel via Web:

A basic admin interface could be designed using NodeMCU and hosted on local Wi-Fi to review logs without removing the SD card.

# Conclusion

The Smart Doorstep Delivery Authentication System (SDDAS) successfully addresses a common and growing concern in modern urban life: the security of doorstep deliveries. Through intelligent integration of sensors, RFID authentication, dual camera imaging, and cloud-enabled alerts, the system provides a comprehensive, low-cost solution to monitor, authenticate, and secure parcel deliveries.

The project stands out not only in functionality but also in its robustness and modularity. With multiple fallback systems (SD logging, image proof, and dual-stage verification), SDDACS ensures traceability and immediate response to suspicious activity. While certain limitations exist, they open up future scopes for innovation and enhancements.

The project also served as a practical and academic exercise in embedded system design, cybersecurity fundamentals, IoT integration, and teamwork. It lays the foundation for broader applications, such as secure package lockers in apartments or smart logistics access control.  
  
  
References

Ai-Thinker, n.d. ESP32-CAM Product Specification. [pdf] Available at: https://loboris.eu/ESP32/ESP32-CAM%20Product%20Specification.pdf [Accessed 9 Apr. 2025].

Arduino, n.d. MFRC522. [online] Arduino Documentation. Available at: https://docs.arduino.cc/libraries/mfrc522/ [Accessed 9 Apr. 2025].

Necula, B., n.d. HX711 Arduino Library. [online] Arduino Libraries. Available at: https://docs.arduino.cc/libraries/hx711-arduino-library/ [Accessed 9 Apr. 2025].

Lough, B., n.d. Universal Telegram Bot Library. [online] Arduino Libraries. Available at: https://docs.arduino.cc/libraries/universaltelegrambot/ [Accessed 9 Apr. 2025].

Telegram, n.d. Telegram Bot API. [online] Available at: https://core.telegram.org/bots/api [Accessed 9 Apr. 2025].

Components101, n.d. ESP32-CAM Camera Module Pinout, Datasheet, Features and Specs. [online] Available at: https://components101.com/modules/esp32-cam-camera-module [Accessed 9 Apr. 2025].

Balboa, M., n.d. Arduino RFID Library for MFRC522. [online] GitHub. Available at: https://github.com/miguelbalboa/rfid [Accessed 9 Apr. 2025].

Tillaart, R., n.d. HX711 Arduino Library. [online] Arduino Libraries. Available at: https://docs.arduino.cc/libraries/hx711/ [Accessed 9 Apr. 2025].

Lough, B., n.d. Universal-Arduino-Telegram-Bot. [online] GitHub. Available at: https://github.com/witnessmenow/Universal-Arduino-Telegram-Bot [Accessed 9 Apr. 2025].

Telegram, n.d. Bots: An Introduction for Developers. [online] Available at: https://core.telegram.org/bots [Accessed 9 Apr. 2025].