CareNest: Where Love Meets Technology

(Smart cradle monitoring system)

24-FYP-203

Session 2021-2025

BACHELOR OF SCIENCE
IN
SOFTWARE ENGINEERING

SUBMITTED BY

Fasih Ahmad khan	21-NTU-CS-1307
Hashim bin Hafeez	21-NTU-CS-1333
Ali Hassan	21-NTU-CS-1297

SUPERVISED BY

Mr. Abdul Qadeer

CO-SUPERVISED BY

Miss Sana Ikram

Department of Computer Science National Textile University, Faisalabad-37610, Pakistan

DECLARATION

We hereby declare that this project report entitled "**CareNest (Smart cradle monitoring system)**" is written by us and is our effort and that no part has been copied or taken without a mentioning reference to the source.

Group Members:

Mr. Fasih Ahmad Khan
21-NTU-CS-1307
E-mail: fasihkhan124124@gmail.com

_____ _____
Signature Date

Mr. Hashim bin Hafeez
21-NTU-CS-1333
E-mail: uncle.hash545@gmail.com

_____ _____
Signature Date

Mr. Ali Hassan
21-NTU-CS-1297
E-mail: Ali@gmail.com

_____ _____
Signature Date

CERTIFICATION

This project written by Fasih Ahmad khan(21-NTU-CS-1307), Hashim bin Hafeez(21-NTU-CS-1333), and Ali Hassan(21-NTU-CS-1297) under the direction of their supervisors and approved by all the members of the FYP committee, has been presented to and accepted by the Department of Computer Science, in the partial fulfillment of the requirement of the degree of Bachelor of Science in Software Engineering.

Supervisor Mr. Abdul Qadeer	Date
Co-Supervisor Miss Sana Ikram	Date
Internal Examiner	Date
External Examiner	Date
FYP-Convener Mr. Muhammad Shahid	Date
Head of Department Mr. Muhammad Asif	Date

Acknowledgement

First of all, In the Name of **Allah**, the Most Beneficent, the Most Merciful. All the praises and thanks be to Allah. A lot of love for our beloved **Holy Prophet MUHAMMAD (S.A.W)**, his guidance always helps us to get the right path. After that, we also proudly declare that the success parameters of this project are the endless prayers and support of our parents.

We would also like to thank our supervisor, **Mr. Abdul Qadeer**, for his assistance with the project, particularly with the paperwork. Every time we meet with him, he is always ready to help us complete our project and documentation. We also wish to thank everyone who assisted us at various phases of the project.

We extend our thanks to our co-supervisor**, Miss Sana Ikram**, for her exceptional management skills. Her ability to simplify our tasks which has significantly improved our efficiency and comprehensiveness in work

We cannot miss the assistance of the **DCS services department**. Special thanks to the DCS services crew for providing us with a location to sit and work in the DCS department, as well as on-demand help. Their helpful attitude was exceptional, and it contributed significantly to the Experts' overall success.

Abstract

The CareNest is a smart cradle monitoring system designed to assist parents in caring for their newborns while managing other tasks. It uses multiple sensors to monitor vital parameters such as baby presence, body temperature, sound levels, and environmental conditions. The system provides real-time notifications in case of anomalies and offers live video streaming for remote monitoring. With an insightful mobile interface, CareNest ensures continuous infant safety and convenience for busy caregivers.

Table of Contents

List of Tables

Chapter 2

Chapter 3

Chapter 7

List of Figures

List of Abbreviations

IOT	Internet of Things
UI/UX	User Interface / User Experience
API	Air Quality Index
SDK	Software Development Kit
AQI	Air Quality Index
ESP	Electronic Stability Program (ESP32, ESP8266 – Microcontrollers)
HTTP	Hypertext Transfer Protocol
JSON	JavaScript Object Notation
IDE	Integrated Development Environment
UML[1]	Unified Modeling Language
DFD	Data Flow Diagram
3-Tier	Three-Tier Architecture
NoSQL	Non-Relational Database

Chapter 1: Introduction

1.1: Problem Statement:

Parents face difficulty in continuously supervising their newborns, leading to stress, reduced productivity, and potential health and safety risks for both parent and child.

1.1.1: Challenges in Infant Monitoring

Supervising and monitoring an infant is a major challenge in the life of a parent. Parents often find it difficult to supervise their newborn child for the whole day. On the other hand, a newborn child requires proper attention and supervision to make sure that they are healthy and safe. This challenge in parenting life is unavoidable as even a stay-at-home parent cannot attend to the infant for 24 hours. This challenge leads to the following problems in the daily life of a parent:

- **Neglecting other tasks:**
  An infant-attending parent is more likely to miss the other tasks assigned to them as only this task requires 100 percent attention of them [2].
- **Compromised Health:**
  This challenge poses risk to the health of both parent and child [3] as doing supervision for a whole day can be stressful for the parent and not doing it properly can cause issues with the infant.
- **Financial Strain:**
  Attending an infant may hinder one's financial activities. Hiring a babysitter is also a financial burden on a parent's account.

1.2: Proposed Solution:

We propose to design a CareNest (Smart cradle monitoring system) that overcomes all the problems faced by the parents. The system would utilize advanced technologies such as IoT [4] for real-time video and audio monitoring, alerts, and monitoring environmental factors. Major features of the system include:

- **Video streaming:** Real-time visual of the baby's activities.
- **Smart alerts**: Alert generation on time to events. the system will also enable alerts management
- **Baby temperature and humidity monitoring:** Maintain the baby environment
- **Mobile application:** for the user-friendly interface to control and monitor.

Providing these features, our solution will help parents remain connected to the baby and allow them to relax knowing the baby is in good care when they are away.

1.3: Purpose:

The main purpose of this project is to solve daily life problems faced by parents that they face while caring for their child. It aims to enable parents to keep an eye on the infant remotely and to make sure that their infant is safe while they are not attending physically.

1.4: Project Goal:

To develop a reliable and user-friendly remote baby monitoring system that enhances parental peace of mind and ensures the safety and well-being of infants.

1.5: Project Objectives:

The objectives of this project are as follows:

1.5.1: Hardware Development:

- Design and develop a robust hardware system, including a camera, sensors, and a processing unit.
- Implement secure wireless connectivity for reliable data transmission.

1.5.2: Software Development:

- Develop a user-friendly mobile application.
- Implement real-time video streaming.
- Ensure secure data transmission and storage.

1.5.3: User Interface Design:

- Create a visually appealing user interface for the mobile app.
- Design clear and concise notifications to alert parents of important events.

1.6: Project Scope:

The scope of this project is as follows:

1.6.1: Target Audience:

This project initially targets one person with parenting responsibilities (Parents, Caretakers, Babysitters, Nannies).

1.6.2: Features:

This project includes core functionalities like video camera surveillance, temperature detection for fever alerts, noise detection for cry alerts, audio content for calming the infant, and humidity detection for ensuring a healthy environment. This project deals with one baby in the cradle at a time and provides monitoring of one baby only at a time. The project aims to deliver the prototype of deploy sensor on baby cradle (it does not include the manufacturing for PCB board or any other related activity) and the Flutter android application with firebase as a backend.

1.6.3: Scalability:

This system is designed with scalability in mind, allowing for future expansion to monitor multiple cradles simultaneously. This feature makes it suitable for deployment in various settings, including hospital nurseries and daycare centers.

1.7: Proposed Tools and Technologies:

Following tools and technologies are proposed for development of this project:

1.7.1: Software:

1. **Figma** – Used for designing the user interface (UI) and user experience (UX) of the mobile application.
2. **Android Studio** – IDE used to manage and run the Flutter-based mobile application on Android devices.
3. **Flutter** – Cross-platform development framework used to build the mobile application with a single codebase.
4. **Dart** – Programming language used with Flutter to implement the app's logic and UI.
5. **Firebase** – Backend-as-a-Service (BaaS) platform used for real-time database, authentication, and notifications.

1.7.2: Hardware:

1. **PlatformIO + VSCode** – Development environment for writing and debugging embedded system code for IoT devices.
2. **Microcontroller**:
    - **ESP32–** The core microcontroller unit that processes data from sensors and modules, enabling IoT functionality.

3. **Sensors and Modules**:
    - **Weight Measurement:** HX-711 module with load cells (15kg capacity) for monitoring the baby's weight in the cradle.
    - **Temperature Monitoring**: DS18B20 (Probe) – Measures the ambient temperature with high accuracy.
    - **Noise Detection**: MAX9814 microphone module to monitor noise levels around the baby.
    - **Air Quality Index (AQI)**: MQ-135 sensor to monitor air quality and detect harmful gases.
    - **Presence Detection**: ultrasonic HC-SR04 to detect the presence of baby in the cradle
4. **Audio System**:
    - **DFPlayer Mini** – A compact MP3 player module for playing lullabies and soothing sounds.
    - **8-ohm Speaker** – Outputs the audio from the MP3 player module.
5. **Camera**:
    - **ESP32-CAM** – A camera module integrated with the ESP32 microcontroller to provide live video streaming of the baby in the cradle.

1.8: Project Scheduling:

We use Gantt Chart [5] to show the overall schedule of the project, the starting and ending time and time duration, each activity takes. So, here is the Timeline of the project shown by a Gantt chart in table

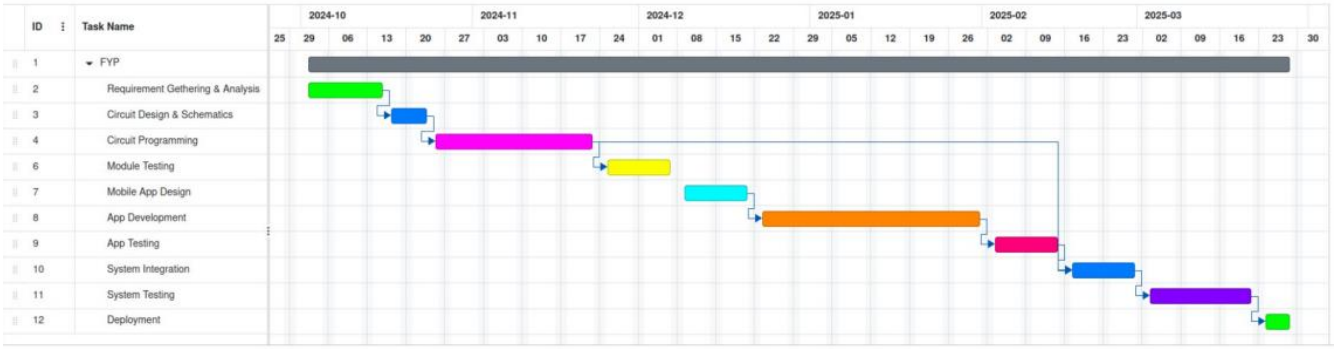


Figure 1. 1 Gantt Chart

Chapter 2: Literature Overview

This chapter will undertake a critical review of the existing systems and technologies related to our project. We will analyze applications currently in use by a wide range of end-users and stakeholders, highlighting their limitations and shortcomings. Analyzing these existing solutions, we will be able to identify the essential features lacking in current offerings and which are thus critical for our proposed system to effectively meet the identified needs.

2.1: Related Work:

The systems on the market currently are mostly environment centric. These systems help parents to smartly monitor their baby, but they are not very effective as their primary focus is to monitor the environment in which the infant is residing rather than focusing on the newborn. They share various common traits with our system, but they have various drawbacks too that are being addressed by our system. However, here we will dig into discussion of such two systems.

2.1.1: VTech Upgraded Smart Wi-Fi Baby Monitor

Owned by V-Tech [6] store, this system is labeled as a baby monitoring system but is primarily developed to be an indoor security system for households. It has its own monitoring screen that relates to a camera with two-way audio communication, Temperature sensors for environment not for baby temperature, Lullaby activation, night vision, etc. It's a plug and play system and just includes a monitor and a camera.

2.1.2: Ellie Smart Baby Monitor

Developed by EllieHello [7], this system is a smart baby monitoring system which allows features like live video streaming, two-way audio communication, mobile application, night vision, etc. This product is widely used for infant monitoring, and it has a plus point of having a mobile app that eliminates the need for a dedicated screen.

Comparison with existing technologies:

Table 2. 1 Comparison between the existing technologies and our system.

Features	**Technologies**		
	Vtech Baby Monitor	**Ellie Smart Monitor**	**Our Work**
Temperature Sensor	Measures Temperature of the Environment	Measures Temperature of the Environment	Measures Temperature of the Infant
Noise Detector	Yes	Yes	Yes
Air Quality Check	No	No	Yes

Spoiled Diaper Detection	No	No	Yes
Live Stream	Yes	Yes	Yes
Play Lullabies	Yes, but not automated	Yes, and automated	Yes, and automated
Baby Presence Detection	No	No	Yes
Mobile Application	No	Yes	Yes
Notification Alerts	No	Yes	Yes

2.2: Reasons to Develop:

There are numerous reasons that have led to the development of this system. A major factor is limitations in the previously existing systems coupled with increasing necessity.

The following are the reasons for the development of this system:

- The technologies mentioned above are more focused on the environment [3] than they are on the infant. This factor molds them into more surveillance systems than baby monitoring systems. The core focus of our system is to monitor the infant's state rather than the environment, ensuring the best possible care for the baby.
- Most baby monitoring systems available on the market require users to purchase a special monitor screen that interacts with the system's features. Our system offers application access to these features, installable on the user's mobile phone. This eliminates the need for extra equipment, making it simpler, more efficient, and cost-effective.
- The user interface of our application is simple and intuitive, allowing even naive users to easily operate it without any prior training.

In short, we are designing this system to prioritize infant welfare by focusing on their state instead of just monitoring the environment and providing a user-friendly mobile application that does not require extra hardware.

2.3: Sensor Parameters and Threshold Values

To ensure accurate, safe, and meaningful monitoring of the infant’s condition, each sensor integrated into the CareNest system is configured with scientifically supported threshold values. These parameters were determined based on sensor datasheets, healthcare guidelines, and prior research studies in the field of IoT-based health monitoring

Table 2. 2 Parameters and Threshold Values for the Sensors

Sensor	Measured Parameter	Threshold Value	Purpose
Temperature Sensor	Infant body temperature [8]	Normal: 97.7–99.5°F Fever: >100.4°F	Detect fever or hypothermia
Air Quality Sensor	Gas concentration / AQI [9]	AQI value > 200 = Poor air quality	Alert for unsafe breathing environment

Chapter 3: System Requirements

This document outlines the detailed system requirements for the CareNest (Smart cradle monitoring system) application. It defines the functional and non-functional requirements that the system must meet to satisfy the business needs. These requirements serve as the foundation for the system's design, development, and testing phases.

3.1: Functional Requirements:

The following are the functional requirements for the system:

3.1.1: User Signup

3.1.1.1: User Registration:

The system should allow new users to create accounts by providing the following information:

- Email address
- Password

3.1.1.2: Password Strength:

The system shall enforce password strength requirements, including minimum length of 8 characters, and unique character usage.

3.1.2: User Login

3.1.2.1: Login Credentials:

The system allow users to log in using their registered username or email address and password.

3.1.3: Temperature Monitoring

3.1.3.1: Real-time Temperature Display:

The system shall display the current temperature of the baby's body in real-time on the user interface.

3.1.3.2: Temperature Alert:

The system shall send an alert notification to the user if the temperature exceeds or falls below predefined Health medical thresholds.

3.1.4: Air Quality Index Monitoring

3.1.4.1: AQI Display:

The system shall display the current air quality index altogether based on presence of hazardous gases like Carbon Monoxide (CO), Nitrogen Oxides (NOx), Ammonia (NH3), and Volatile Organic Compounds (VOCs). Ensuring safe air for the baby.

3.1.4.2: AQI Alert:

The system shall send an alert notification to the user if the AQI reaches an unhealthy level.

3.1.5: Weight Monitoring

3.1.5.1: Baby Presence Detection:

The system shall detect the presence of the baby based on weight changes and temperature fluctuation on the cradle's sensor.

3.1.6: Lullaby Speaker Control

3.1.6.1: Lullaby Playback:

The system shall allow the user to play automatically lullabies through the cradle's speaker.

3.1.7: Live Camera Streaming

3.1.7.1: Video Streaming:

The system shall provide a live video feed from the cradle's camera to the user's device.

3.1.7.2: Secure Access:

The system shall implement secure authentication and authorization mechanisms to restrict access to the video feed to authorized users.

3.1.8: Notification System

3.1.8.1: Alerts:

The system shall send notifications to the user's mobile device for any anomalies in temperature, AQI, weight, or humidity.

3.1.8.2: Customizable Notifications:

The system shall allow users to customize notification settings, including:

- Enable / Disable different Types of alerts

3.1.9: Mobile Application Access

3.1.9.1: User-Friendly Interface:

The mobile app will provide a user-friendly interface for accessing all monitoring features and controlling the cradle's functionalities.

3.1.10: Secure Data Handling

3.1.10.1: Data Encryption:

The system shall encrypt all sensitive data, including personal information and health data, during transmission and storage.

3.1.10.2: Secure Authentication:

The mobile app shall implement strong authentication mechanisms, such as password-based authentication.

3.1.10.3: Data Privacy:

The system shall comply with relevant data privacy regulations and protect user data from unauthorized access.

3.1.11: Diaper Changing alert

3.1.11.1: Change Alert:

The system shall send an alert notification to the user when the diaper is contaminated and needs changing.

3.2: Non-Functional Requirements:

These are considered to be prerequisites for performance. Usually, these are not required by users but provided to satisfy investors and meet standard quality

3.2.1: Usability

The system's interface should be easy to use.

3.2.2: System Performance

The system should provide timely notifications and updates and operate with efficiency.

3.2.3: Security

The system should provide appropriate safety measures to protect user data.

3.2.4: Availability

Ensure that there is little downtime for upgrades or maintenance and that the system is available and functioning around-the-clock.

3.2.5: Reliability

The system should be trustworthy, precisely detecting baby environment and generating accurate data.

3.3: Use Case Diagram:

The most suitable method for visualizing how actors interacts with system components is to create use case diagrams, which visually depict which actors can access or perform which system functionality. Figure 3.1 shows the use case diagram for the system:

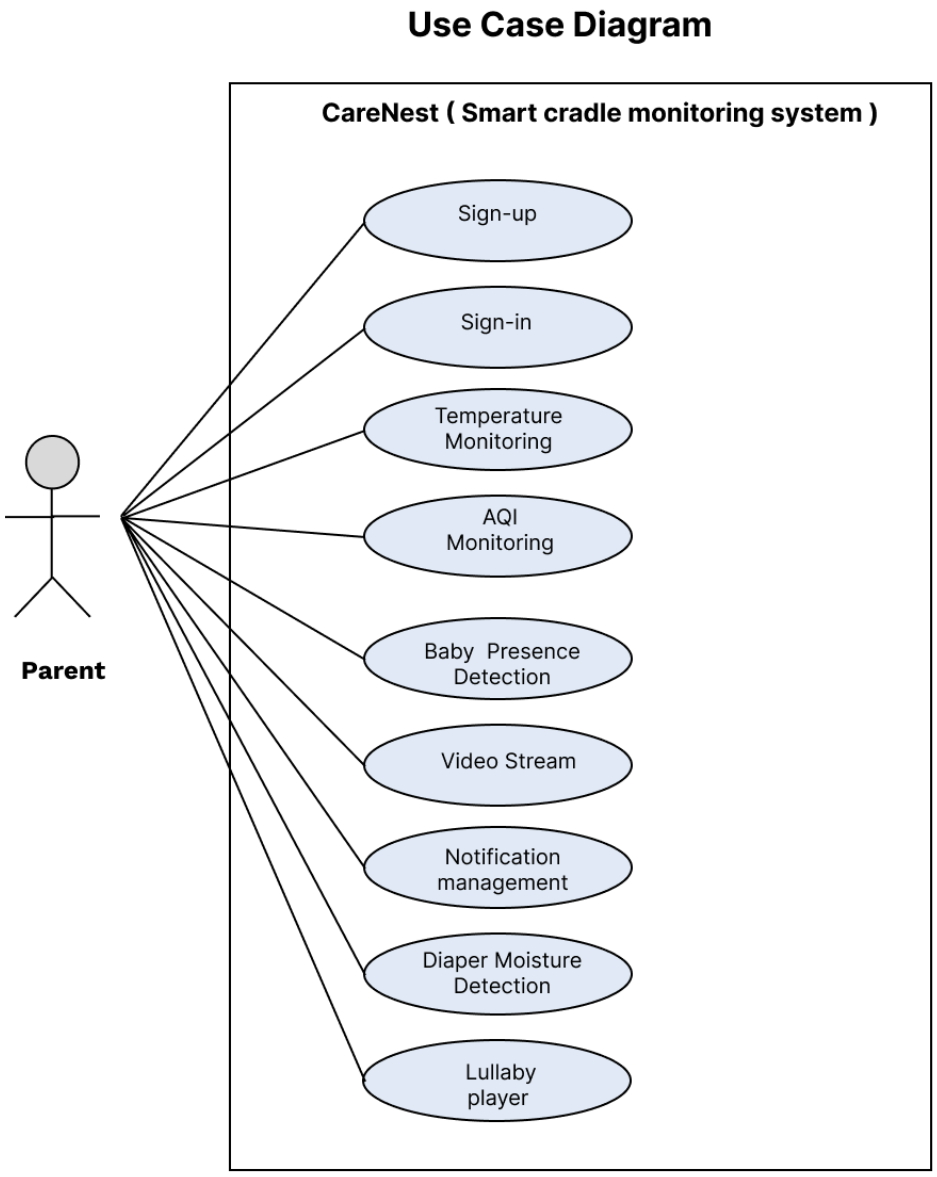


Figure 2. 1 Use Case Diagram of CareNest

Use Case Description:

Table 3. 1 Sign-in use case description

Field	Description
Use case Id	1
Use Case Name	Sign-in
Actor	Parents (Primary)
Use case Description	This use case describes the process by which a parent (user) signs into the CareNest system using valid credentials to access features such as monitoring notifications, live video, and cradle statistics.
Pre-Condition	• The parent must have an existing account with valid credentials (email/username and password). • The application must be connected to the internet.
Normal Flow	1. The parent launches the CareNest application. 2. The parent selects the Sign In option on the home screen. 3. The system displays the Sign in form with fields for email/username and password. 4. The parent enters their email/username and password. 5. The parent submits the form by clicking the Sign In button. 6. The system sends the entered credentials to the authentication system for validation.
Post-Condition	• The parent is successfully authenticated.

	• The parent has access to the application features (e.g., monitoring notifications, viewing data).
Exception	• Invalid Credentials • Account Not Found • Network Issues

Table 3. 2 Sign-Up use case description

Field	**Description**
Use case Id	2
Use Case Name	Sign-up
Actor	Parents (Primary)
Use case Description	This use case describes the process by which a parent creates an account on the CareNest system to access the application's features, such as baby monitoring, notifications, and cradle statistics.
Pre-Condition	• The parent must have the CareNest application installed • The application must be connected to the internet.
Normal Flow	1. The parent opens the CareNest application. 2. The parent selects the **Sign-Up** option on the home screen. 3. The parent enters the required details and submits the form by clicking the **Sign-Up** button.

	4. The system validates the input. 5. If the input is valid, the system registers the parent in the authentication system.
Post-Condition	• The parent's account is successfully created in the system. • The parent can sign in to access application features
Exception	• Invalid Email Format • Network Issues

Table 3. 3 Temperature Monitoring use case description

Field	**Description**
Use case Id	3
Use Case Name	Temperature Monitoring
Actor	Parents (Primary) Smart cradle (Secondary)
Use case Description	This use case describes how the system monitors the baby's body temperature and environmental temperature in the cradle and sends the data to the mobile application. It ensures real-time temperature monitoring and triggers alerts when thresholds are exceeded.
Pre-Condition	• The cradle must be powered on. • The temperature sensors (IR and Probe) must be calibrated and function correctly. • The mobile application must be connected to the system via the IoT network.

Normal Flow	1. The Ultrasonic sensor detects the baby's presence in the cradle 2. The probe temperature sensor measures the baby's body temperature 3. If the baby's body temperature exceeds or falls below the thresholds, the system generates a temperature alert. 4. Alerts and temperature data are sent to the mobile application.
Post-Condition	• The system continuously monitors the temperature while the baby is in the cradle. • Alerts are successfully sent to the mobile application. • The temperature data is available for real-time monitoring on the mobile app.
Exception	• Sensor Malfunction • Network Failure • Baby Not Detected

Table 3. 4 AQI Monitoring use case description

Field	**Description**
Use case Id	4
Use Case Name	AQI Monitoring
Actor	Parents (Primary) Smart cradle (Secondary)
Use case Description	This use case describes how the system monitors the air quality around the cradle and provides a qualitative

	assessment (e.g., "Good" or "Bad") to ensure a healthy environment for the baby. Alerts are sent if the air quality falls below acceptable levels.
Pre-Condition	• The cradle must be powered on. • The air quality sensor must be functioning properly. • The mobile application must be connected to the system via the IoT network.
Normal Flow	1. The air quality sensor measures the concentrations of gases and other environmental factors in real-time. 2. The system processes the sensor data to calculate the Air Quality Index (AQI). 3. The AQI is translated into a qualitative assessment (e.g., "Good," "Moderate," "Poor"). 4. The system compares the AQI against pre-defined acceptable thresholds. 5. If the AQI assessment is "Poor" or worse, the system generates an alert indicating unhealthy air quality. 6. The qualitative assessment and any generated alerts are sent to the mobile application. 7. The user can view the real-time AQI status and any alerts through the app interface.
Post-Condition	• The system continuously monitors the air quality around the cradle. • Alerts are successfully sent to the mobile application. • The AQI status is available for real-time monitoring on the mobile app.
Exception	• Sensor Malfunction • Network Failure • Inaccurate Data

Table 3. 5 Baby Presence Detecion use case description

Field	Description
Use case Id	5
Use Case Name	Baby Presence Detection
Actor	Parents (Primary) Smart cradle (Secondary)
Use case Description	This use case describes how the system detects the baby's presence in the cradle using the Ultrasonic sensor and weight sensor. It ensures monitoring and related functionalities only activate when the baby is in the cradle.
Pre-Condition	• The cradle must be powered on. • Both the Ultrasonic sensor and weight sensor must be calibrated and function properly. • The system must be connected to the mobile application via the IoT network.
Normal Flow	1. The Ultrasonic sensor detects movement indicating the presence of the baby 2. The weight sensor detects the presence of an object (baby) with a weight within a pre-defined range. 3. The system cross-verifies input from both sensors to confirm the presence of the baby. 4. If both sensors confirm the baby's presence, the system marks the cradle as "occupied." 5. The system activates monitoring features (e.g., temperature, air quality, moisture) once the baby's presence is confirmed. 6. The mobile application receives a notification indicating whether the baby is in the cradle.

	7. The app displays the real-time baby presence status.
Post-Condition	• The system identifies and confirms the baby's presence in the cradle. • Monitoring features are activated if the baby is present. • Notifications regarding the baby's presence status are successfully sent to the mobile application.
Exception	• Sensor Malfunction • Network Failure • Inaccurate Data (false Detection)

Table 3. 6 Video Steam use case description

Field	**Description**
Use case Id	6
Use Case Name	Video Stream
Actor	Parents (Primary) Smart cradle (Secondary)
Use case Description	This use case describes how the system streams real-time video from the cradle's camera to the mobile application, allowing parents or caregivers to monitor the baby remotely.
Pre-Condition	• The cradle must be powered on. • The camera must be installed, configured, and function correctly. • The mobile application must be connected to the system via the IoT network.

	• The user must have a stable internet connection to view the video stream on the mobile application.
Normal Flow	1. The camera starts streaming video when the system is operational and detects the baby's presence in the cradle. 2. The video feed is processed and uploaded to the cloud for secure real-time streaming. 3. The system sends the video stream from the cloud to the connected mobile application. 4. The user accesses the mobile application to view the live video feed of the cradle. 5. The video stream continues as long as the system is active, and the baby is present.
Post-Condition	• The user can view a live video feed of the cradle in the mobile application. • The system ensures secure and uninterrupted streaming as long as conditions are met.
Exception	• Camera Malfunction • Network Failure • Baby Not Present

Table 3. 7 Notification Management use case description

Field	**Description**
Use case Id	7
Use Case Name	Notification management

Actor	Parents (Primary) Smart cradle (Secondary)
Use case Description	This use case describes how the system generates, processes, and delivers real-time notifications to the mobile application for events such as temperature changes, air quality issues, diaper moisture alerts, and baby presence status.
Pre-Condition	The cradle system and mobile application must be powered on and connected via the IoT network. All sensors and monitoring features must be functioning correctly. The user must have enabled notifications in the mobile application settings.
Normal Flow	1. The system detects an event 2. The system generates a relevant notification based on the detected event. 3. The system formats the notification for delivery, including details such as time, event type. 4. The system sends the notification to the mobile application in real-time. 5. The user views the notification on the app and takes action as needed.
Post-Condition	• Notifications are delivered to the user's mobile application in real-time. • The user can access a log of past notifications within the app. • The system ensures that notifications are acknowledged or acted upon where necessary.
Exception	• Sensor Malfunction • Network Failure

Table 3. 8 Diaper Moisture Detection use case description

Field	Description
Use case Id	8
Use Case Name	Diaper Moisture Detection
Actor	Parents (Primary) Smart cradle (Secondary)
Use case Description	This use case describes how the system detects moisture levels in the baby's diaper using a probe moisture sensor and sends an alert to the mobile application when a diaper change is needed.
Pre-Condition	• The cradle must be powered on. • The probe moisture sensor must be installed, calibrated, and function correctly. • The mobile application must be connected to the system via the IoT network.
Normal Flow	1. The moisture sensor continuously measures moisture levels in the baby's diaper. 2. The system compares the detected moisture levels against a pre-defined threshold. 3. If the moisture level exceeds the threshold, the system determines that the diaper needs changing. 4. The system generates an alert indicating that a diaper change is required. 5. The alert is sent to the mobile application in real-time.

Post-Condition	• The system continues to monitor moisture levels in the diaper after the alert is sent. • The user receives diaper-related notifications as needed. • The system ensures accurate tracking of diaper status and minimizes false alerts.
Exception	• Sensor Malfunction • Network Failure • False Positives

Table 3. 9 Play / Pause Lullaby use case description

Field	**Description**
Use case Id	9
Use Case Name	Play / Pause Lullaby
Actor	Parents (Primary)
Use case Description	This use case describes how the system plays or pauses lullabies through the speaker based on user input or detected environmental noise levels.
Pre-Condition	• The cradle must be powered on. • The sound sensor and speaker must be installed and function properly. • The mobile application must be connected to the system via the IoT network.

	• Lullaby files must be available on the speaker's SD card.
Normal Flow	1. The sound sensor continuously monitors environmental noise levels. 2. If noise exceeds a pre-defined threshold, the system automatically plays a lullaby. 3. The lullaby stops automatically after a pre-set duration.
Post-Condition	• The system ensures lullabies are played only when needed or requested. • The mobile app reflects the updated playback status in real-time.
Exception	Speaker Malfunction SD Card Issue Sound Sensor Failure Network Failure

Chapter 4: Methodology

This chapter covers the project lifecycle, which includes feasibility study and detailed project schedule. We will discuss several software development methodologies that are important to the successful delivery of software. This part will review the existing methodologies, the methodology selected for our project, and its implementation flow that will lead to a detailed project schedule. Let us start by discussing the current landscape of existing software development methodologies.

4.1: Methodologies for Software Development:

A software development methodology acts as an outline for developing efficiency within the process of development in a series of well-articulated steps. Productivity improves with this methodology, along with better management of systems. These sets of methodologies, commonly called the SDLC, embrace a set of phases defined by its rules. Two major methodologies can be differentiated: there are two kinds of it, namely, Rapid Application Development, and Plan-Driven. While Waterfall and Spiral models are classed as Plan-Driven, Agile methodologies fall in the category of RAD.

4.1.1: Waterfall Model:

The project uses the Waterfall Model [10], one of the traditional, linear approaches for developing software. It breaks up the project into distinct phases. One of the distinct features of the Waterfall Model is its sequential nature; no phase can be skipped once completed, making it impossible to implement changes to the phase once it is concluded, hence the name "Waterfall" – each phase cascades into the next. In the long run, significant cost and time savings will be achieved if requirements and design are emphasized at the beginning stages of the Waterfall Model.

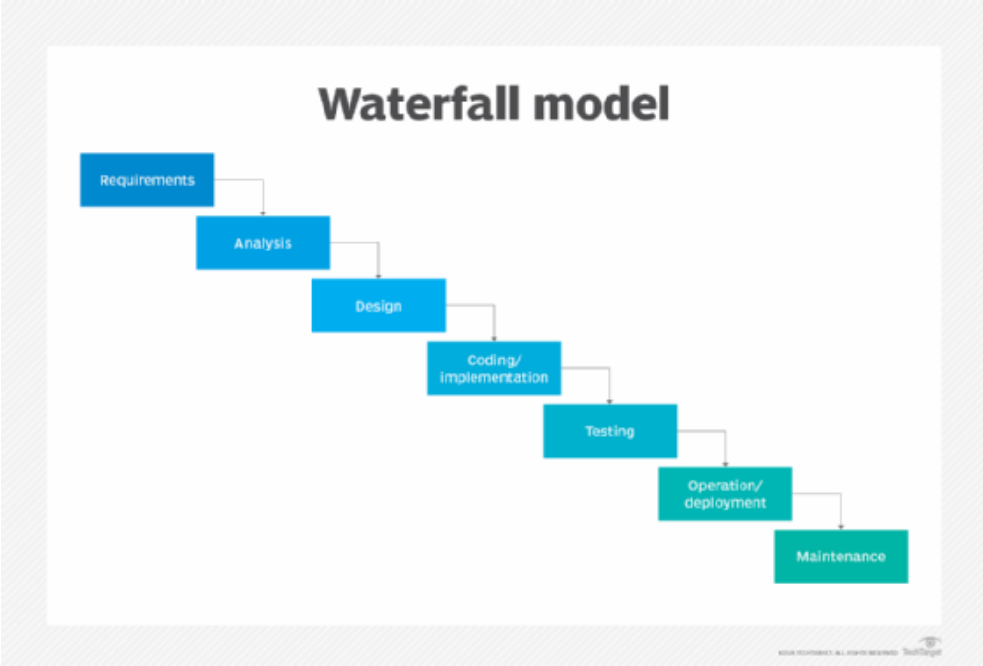


Figure 4. 1 Water Fall Methodology

4.1.2: V Model:

The V-model [11] is a derivative of the Waterfall model emphasizing the strong correlation between the testing activities and the related analysis and design phases. Like Waterfall, it follows the sequential approach, where each stage must be completed before starting the next one. Nevertheless, the V-model is more unique in that it maintains a separate testing phase at every stage of the Waterfall model, which gives it a look like a "V.". Unit and system testing within this framework serve to validate the design of the program, checking whether individual components are correctly functioning and whether the whole system works.

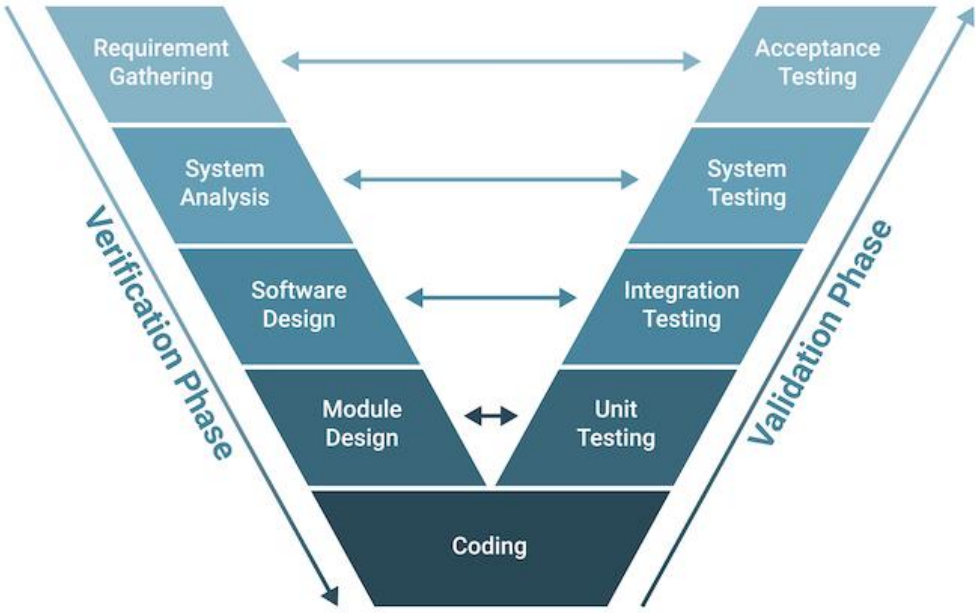


Figure 4. 2 V-model methodology

4.1.3: incremental Model:

The Incremental Model [12] is a software development approach where the big picture of the project is divided into numerous smaller sub-projects that could be easily handled. Dividing the system into groups of functionalities, this module goes through the full life cycle of the process starting from gathering requirements to actual design, implementation, and testing. After that, subsequent releases add incrementally new features and functionalities added to the basic subsystem introduced in the early releases.

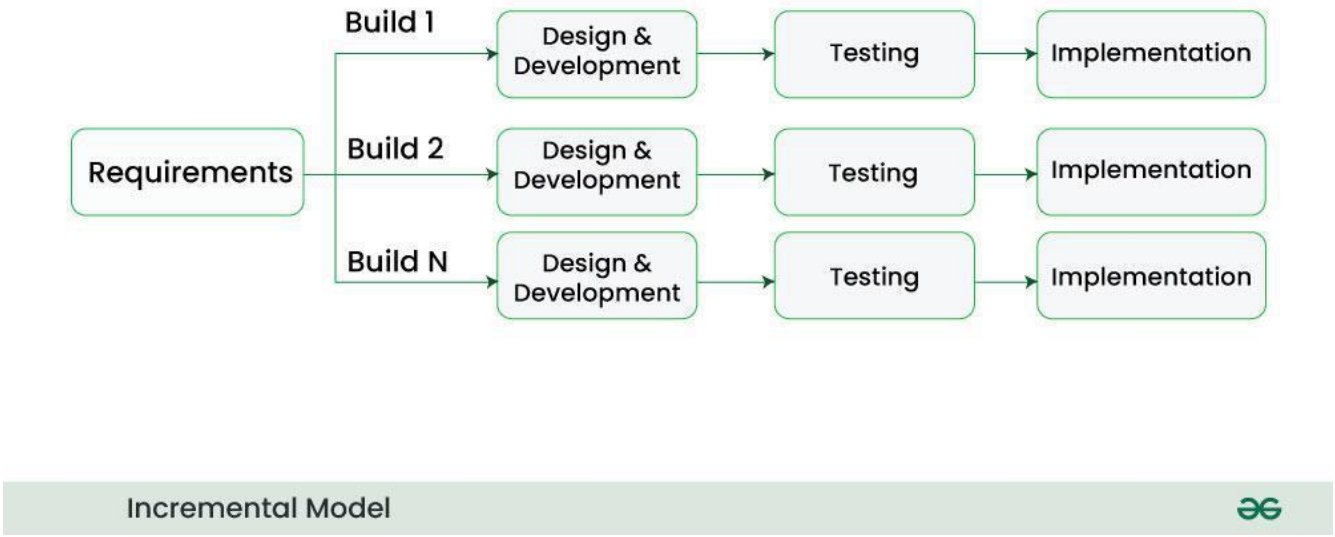


Figure 4. 3 V-model methodology

4.1.4: Iterative Model:

The Iterative Model [13] is a software development approach that breaks the project into smaller cycles known as iterations. Each iteration of the model covers the full lifecycle of software development. Iterative development encourages continuous improvement, giving a working system in the first release and then continues to improve the functionality of each subsystem with each release. This model focuses on continuous testing, which allows project evaluation to be done continuously throughout the project and distributes the work more evenly over the time of the project.

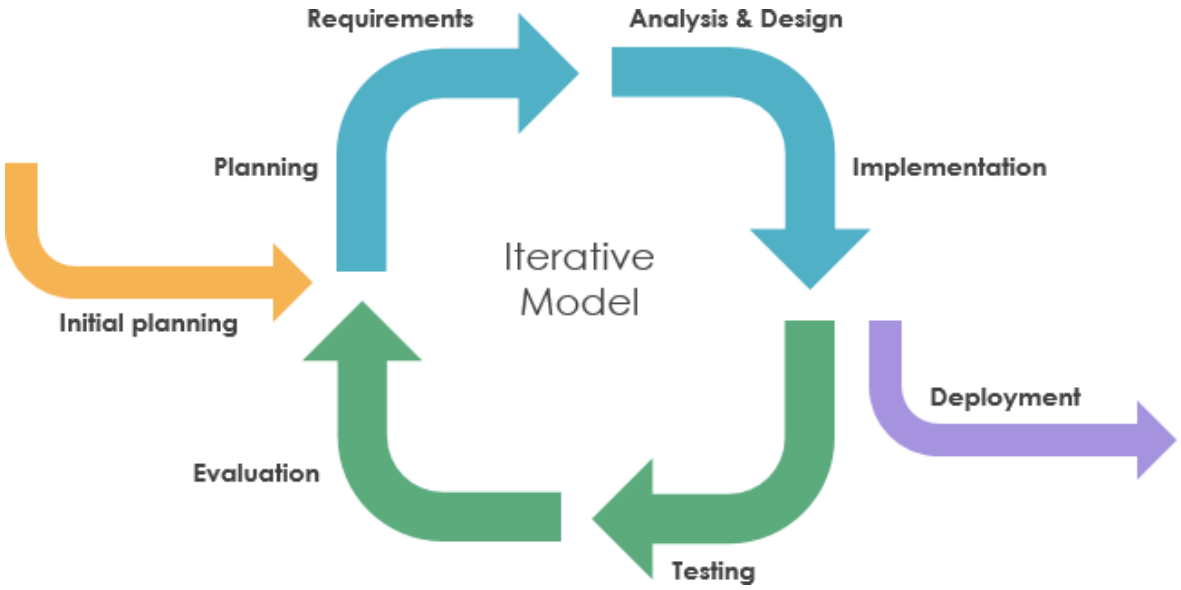


Figure 4. 4 Iterative Methodology

4.1.5: Agile Model:

Agile process models [14] are a software development approach based on iterative development principles. Unlike traditional methods with extensive long-term planning, Agile methodologies break down tasks into smaller, more manageable iterations. While the project scope and initial requirements are established at the outset, Agile emphasizes flexibility. The number of iterations, their duration, and the scope of each iteration are typically defined with greater clarity at the commencement of the development process.

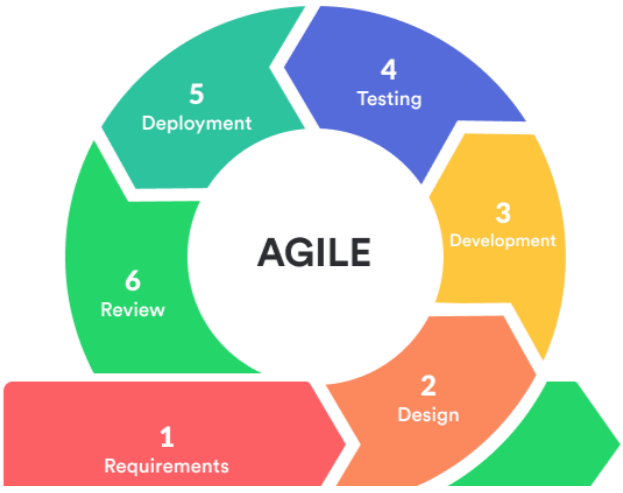


Figure 4. 5 Agile Methodology

4.1.6: RAD Model:

The Rapid Application Development (RAD) model [15] is an incremental software development process that focuses on prototyping and iterative user feedback instead of extensive upfront planning and comprehensive testing. RAD is especially designed to accelerate software delivery within short timeframes with emphasis on user involvement and iterative refinement throughout the development cycle.

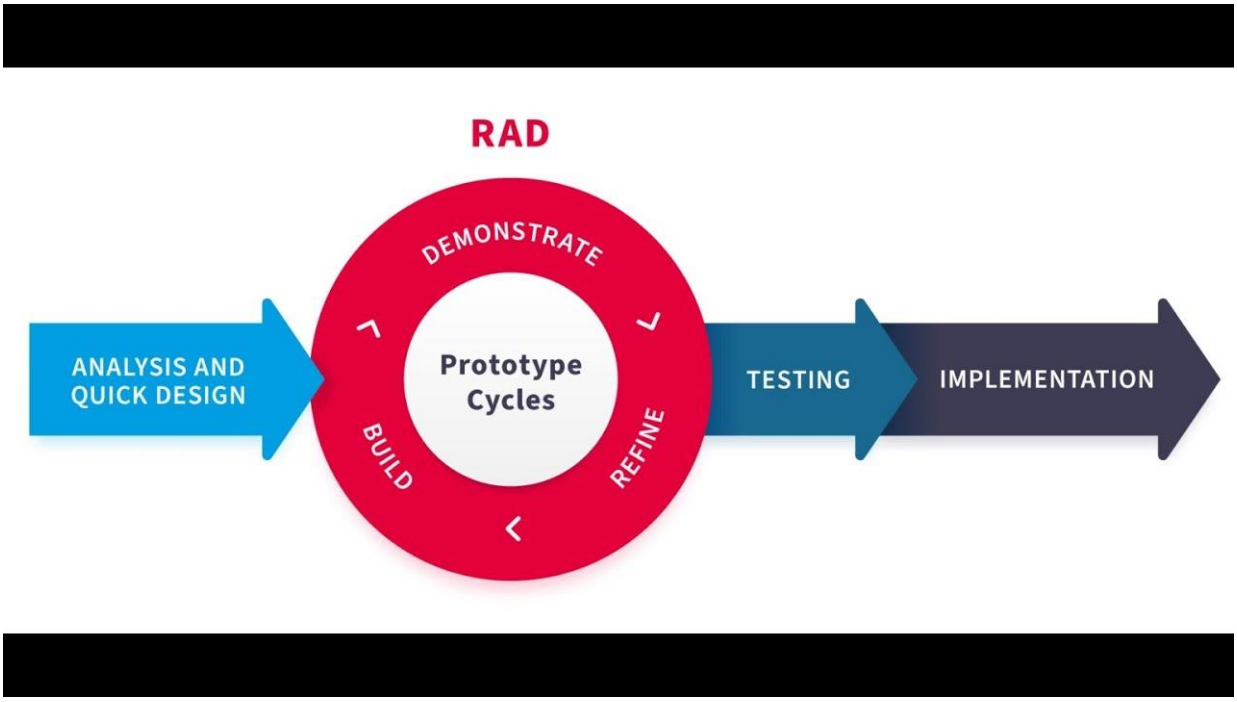


Figure 4. 6 RAD Methodology

4.2: Selected Methodology:

In the current project, we selected the Waterfall Model with combination of iterative model because it provides a clear, structured, and sequential approach to software development. Each phase must be completed before moving on to the next, ensuring thorough documentation, clear planning, and well-defined requirements from the beginning.

4.3: Reasons for Selecting the Methodology:

The following are the most important reasons why we opted for the waterfall method:

4.3.1: Clear Structure and Documentation:

Waterfall ensures that each development phase is completed with proper documentation, making the process more manageable and easier to follow. This structured approach helps maintain consistency and reduces ambiguity throughout the project lifecycle.

4.3.2: Easy to Manage and Track Progress:

Due to its linear nature, the Waterfall model allows for straightforward project tracking. Each completed phase acts as a checkpoint, making it easy to assess progress and identify any delays or issues.

4.3.3: Defined Requirements:

Agile encourages constant communication and collaboration between the stakeholders and end-users during development. More frequent feedback sessions ensure necessary adjustments, thereby leading to continuous improvement towards a final product that is much closer to the needs of the user.

4.3.4: Suitable for Fixed Scope Projects:

In the Waterfall model, all project requirements are gathered and documented at the beginning. This ensures that the development team has a complete understanding of the system before implementation begins, reducing the chances of unexpected changes.

4.4: Conclusion:

The Waterfall software development process is well-suited for this project due to its emphasis on structure, clarity, and predictability. It ensures thorough planning, disciplined execution, and minimizes scope creep making it an appropriate choice for projects with well-defined goals and stable requirements.

Chapter 5: System Architecture

Architecture [16] helps to describe system flexibility that includes process planning and design. It defines the major software components, set of containers and connectors. In this chapter, we will following diagram [17].

5.1: Three Layer Architecture Deployment diagram

This model is widely used for IoT applications and consists of Three Layer Architecture[16] :

1. Perception Layer (Edge Layer - Hardware & Sensors
2. Network Layer (Communication & Cloud)
3. Application Layer (User Interface - Mobile App)

The deployment diagram represents the physical architecture and interactions between different nodes and artifacts that take part in the deployment of the Smart cradle monitoring system application. In this configuration, there are three primary nodes: User Mobile, Firebase Server, and Cradle. The User Mobile node contains the Smart cradle monitoring system App, which is the client-side application that the users interact with. It talks to the Firebase Server using REST APIs, hence carrying out the main functionality of authentication for users and also data management capabilities.

The **Firebase Server** acts as a cloud-based back end hosting all important artifacts, which consist of Firebase Authentication and Firebase API services. They assist in providing the user authentication functionality securely, with an interface on which the mobile app could tap into any available functionalities, like storing data or creating real-time database capabilities.

The third node, **Cradle**, consists of two artifacts: Sensor Firmware and Communication Module. The Cradle node manages sensor data and facilitates communication between the Firebase Server and connected devices. It uses Wi-Fi HTTP [18] protocols to connect with the Firebase Server to enable smooth data transmission and device management.

Overall, below deployment diagram effectively captures the relationships between the nodes and their respective artifacts, showcasing how the Smart cradle monitoring system application operates within this distributed architecture. The interactions among these components ensure that user requests are processed efficiently while maintaining robust communication pathways across the system.

5.2: Activity Diagram:

The workflow of a system with 3 Sensor roles: Temperature Sensor, Noise Sensor, and AQI Sensor is depicted in this activity diagram. When a user logs in successfully, each sensor has unique functionality.

5.1.1: Activity Diagram of CareNest Application UI interaction:

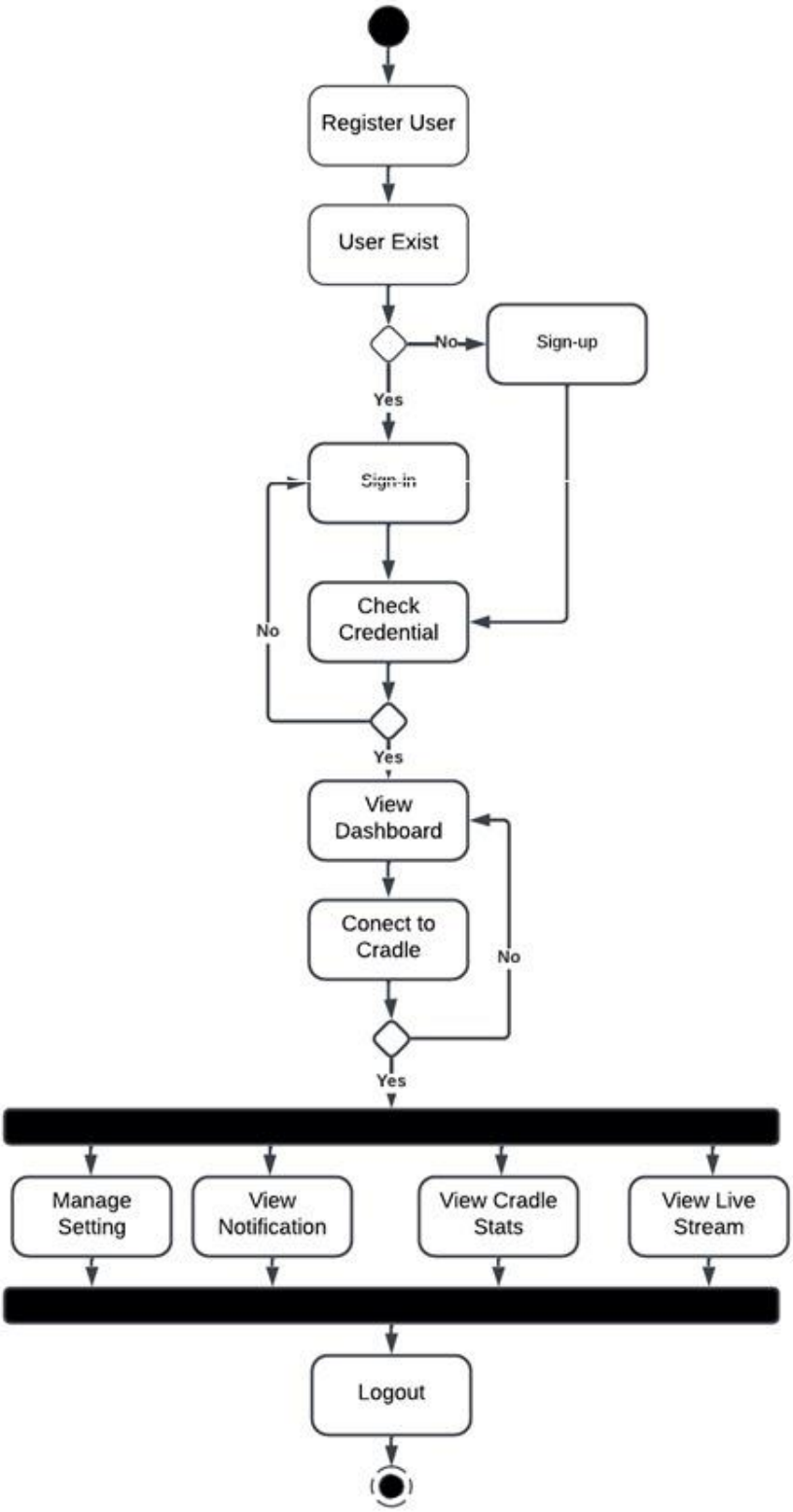


Figure 5. 1 Activity Diagram of CareNest Application

5.1.2: Activity Diagram of Temperature Sensor:

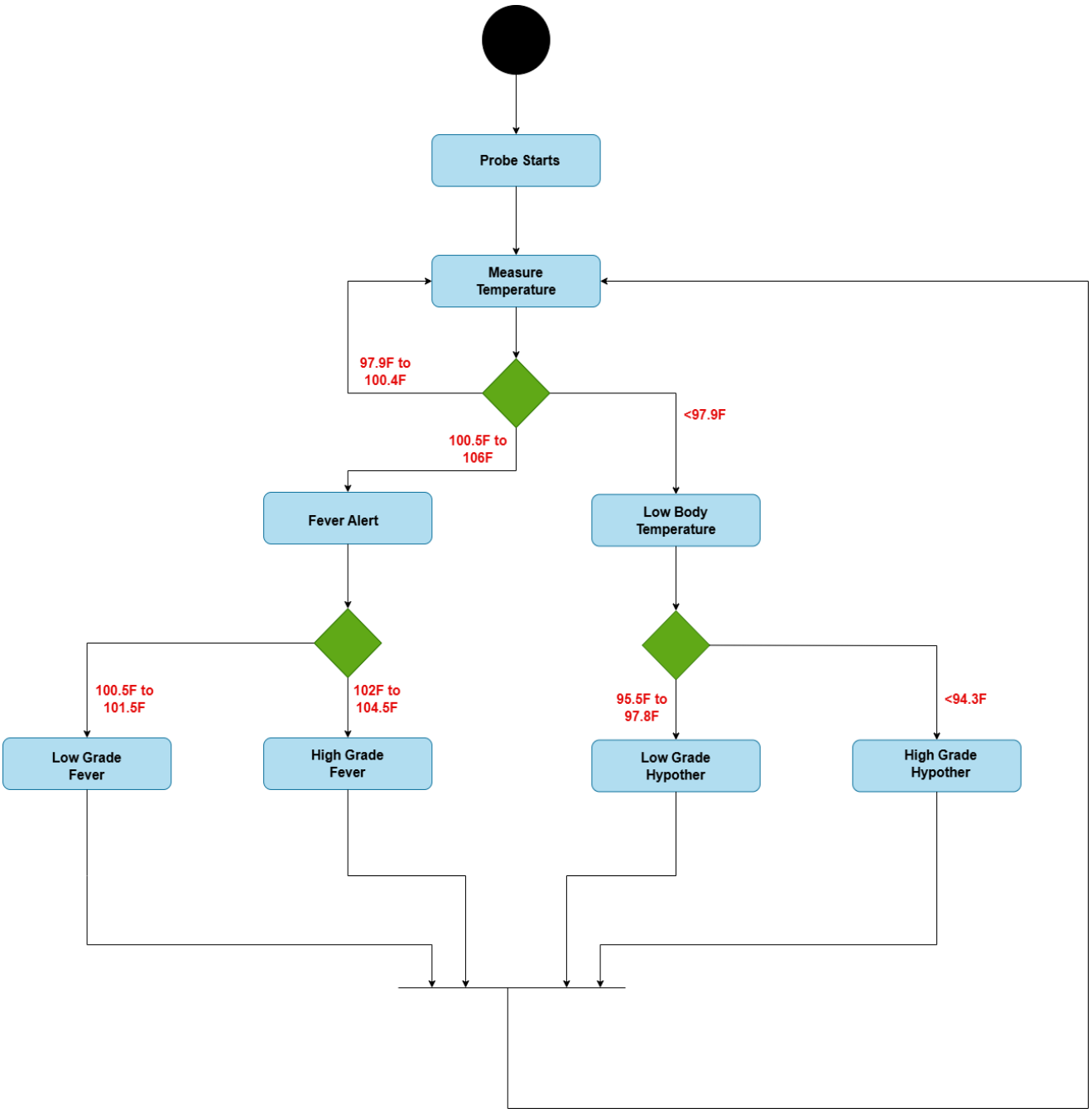


Figure 5. 2 Activity Diagram of Temperature Sensor

5.1.3: Activity Diagram of Noise Sensor:

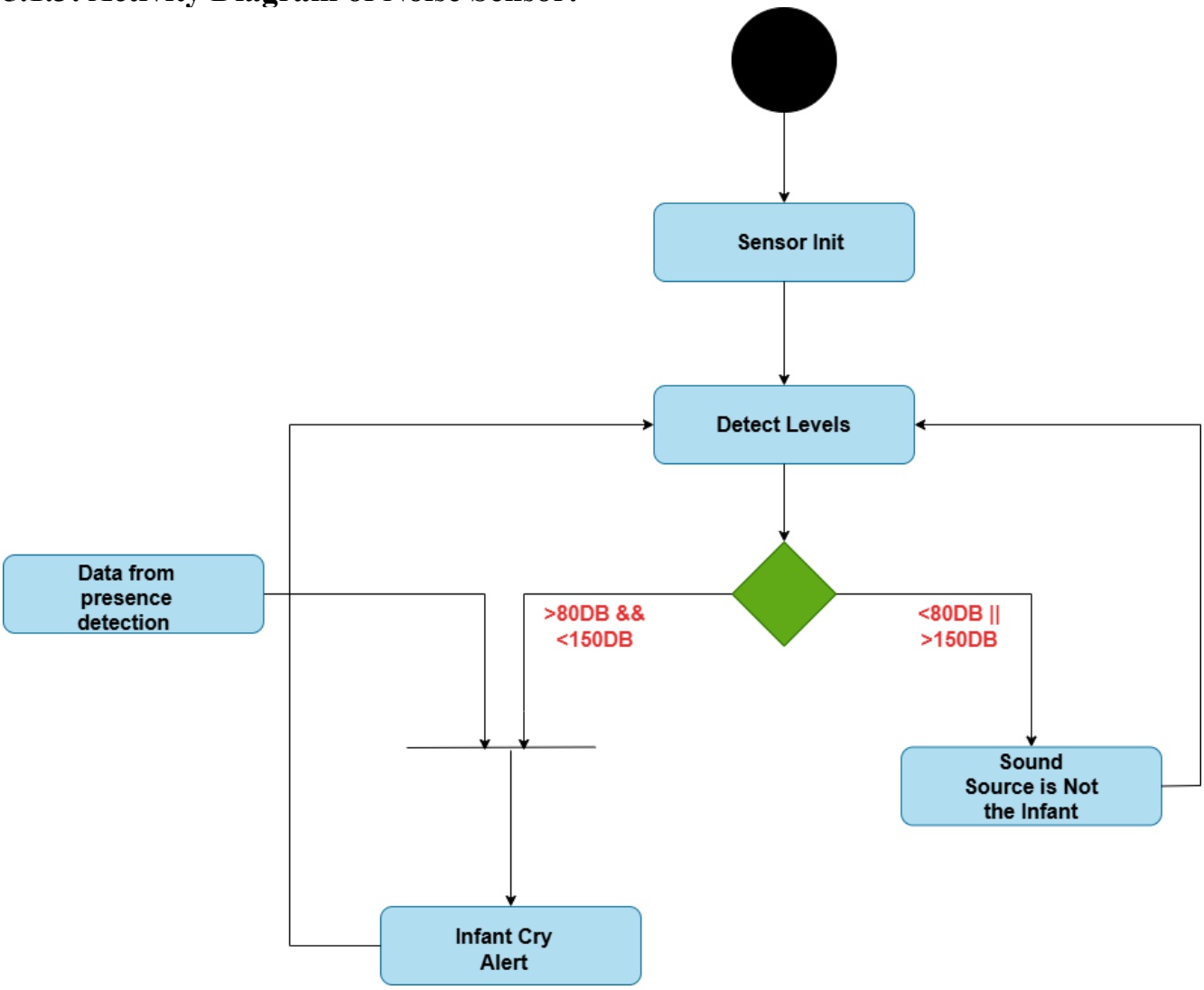


Figure 5. 3 Activity Diagram of Noise Sensor

5.1.4: Activity Diagram of AQI Sensor

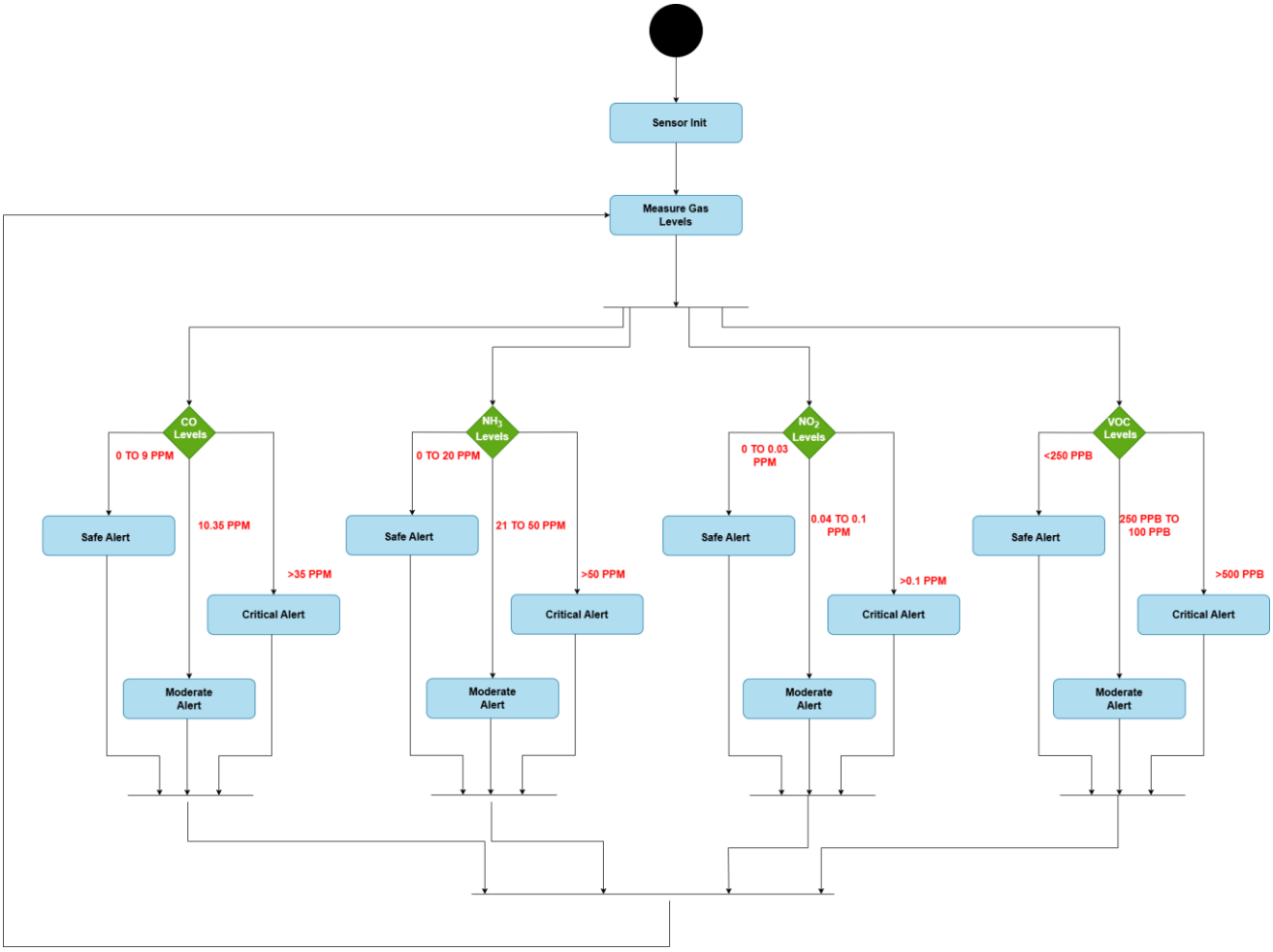


Figure 5. 4 Activity Diagram of AQI Sensor

5.1.5 Activity Diagram of Lullaby Player Including Noise Sensor:

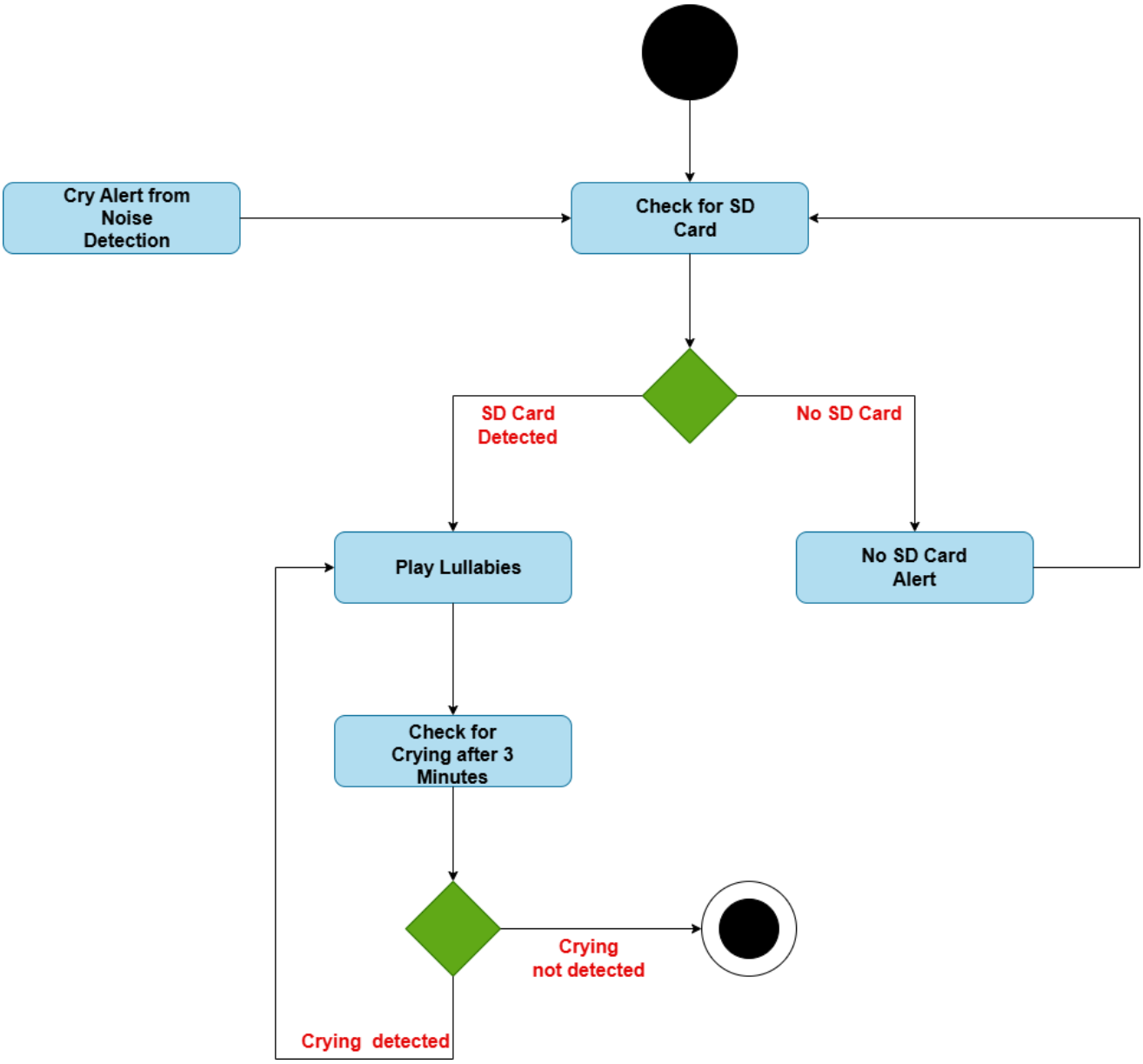


Figure 5. 5 Activity Diagram of Lullaby Player Including Noise Sensor

5.1.6: Activity Diagram of Baby Presence Detection Using Temperature and Weight Sensors:

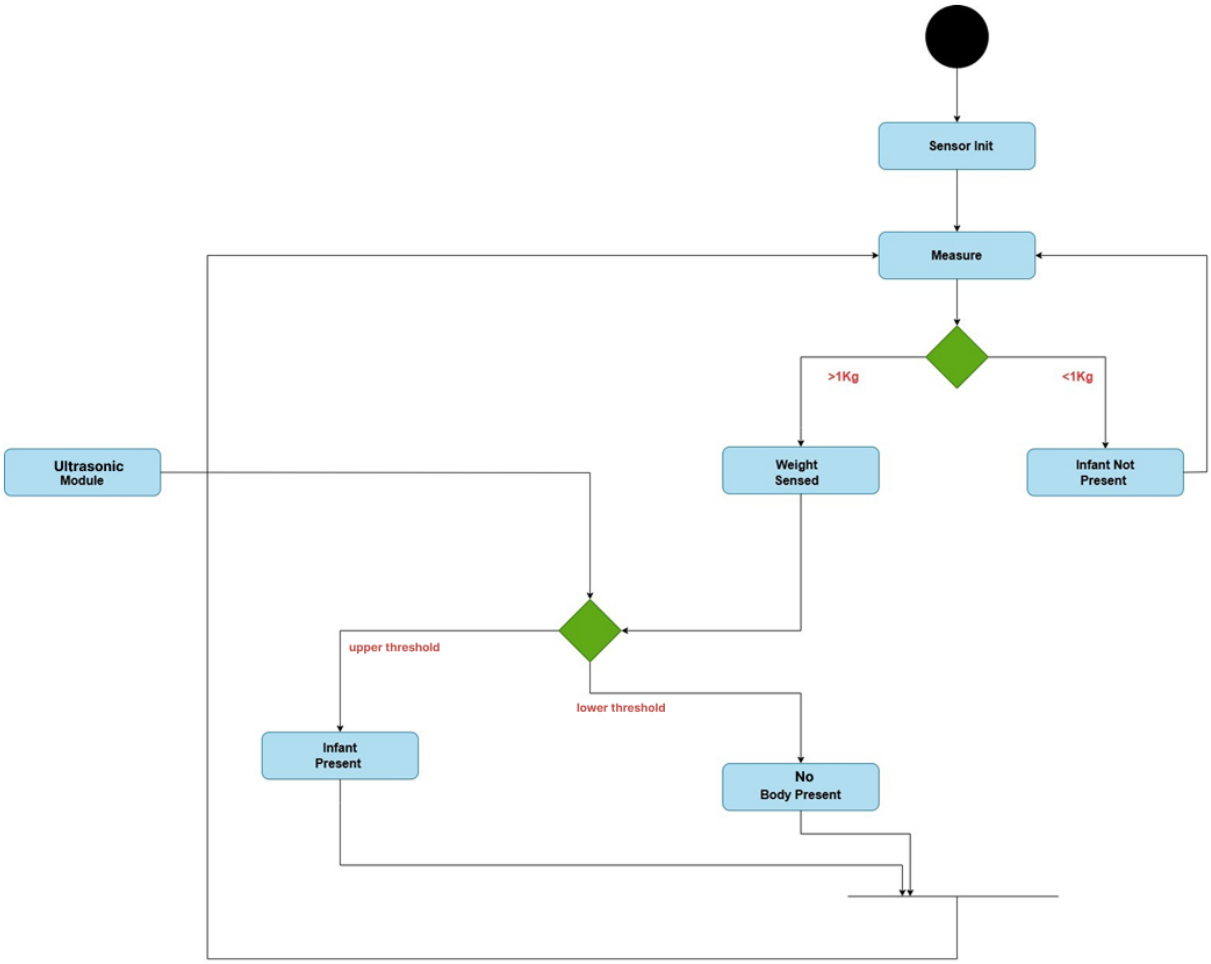


Figure 5. 6 Activity Diagram of Baby Presence Detection

5.3: Data Flow Diagram (DFD):

A Data Flow Diagram (DFD) is a graphical representation of the flow of data within a system. It visually depicts how data moves between processes, data stores, and external entities in a system. DFDs are commonly used to help to illustrate the structure and functionality of a system.

5.3.1: Level 0 Data Flow Diagram

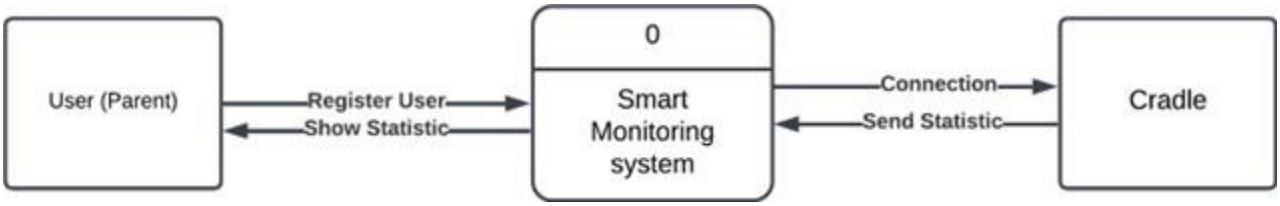


Figure 5. 7 Level 0 Data Flow Diagram

5.3.2: Level 1 Data Flow Diagram

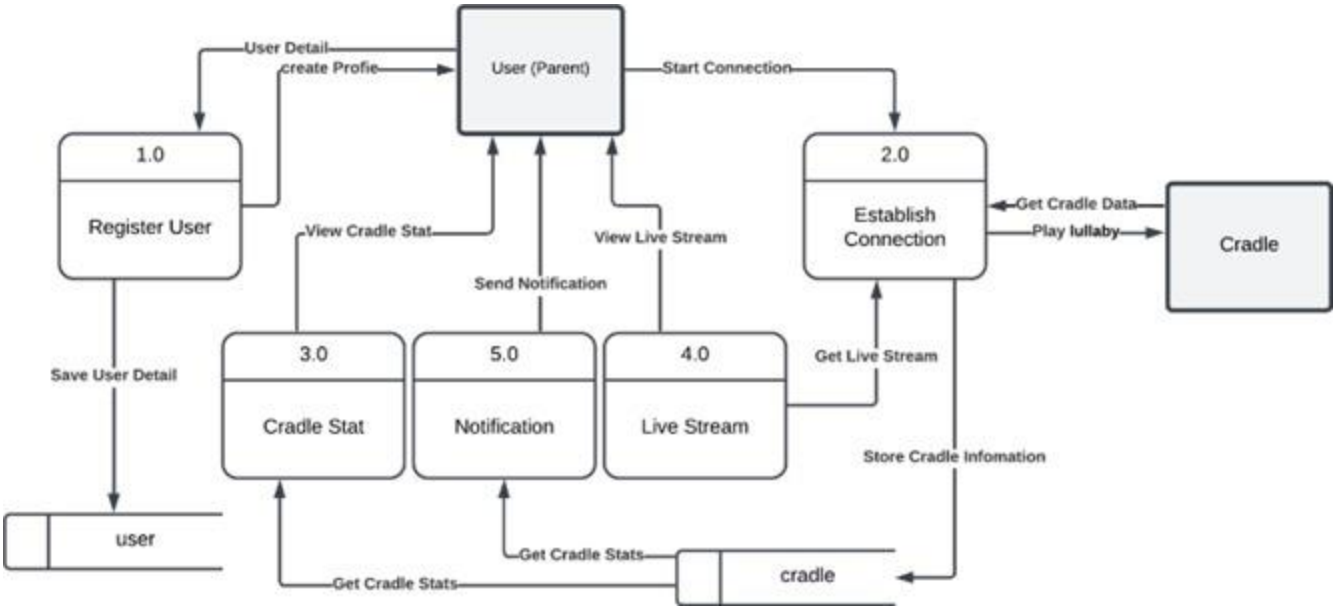


Figure 5. 8 Level 1 Data Flow Diagram

Chapter 6: System Implementation

6.1: System tools and technology

6.1.1: Figma.

Figma is a cloud-based design platform designed for creating user interfaces, prototypes, and collaborative design projects. It allows designers and teams to work together in real time from anywhere in the world. Below are the key features and functionalities of Figma:

- **Real-Time Collaboration:** Figma supports simultaneous collaboration, enabling multiple users to work on the same design file at once. Changes are instantly visible to all team members, promoting efficient teamwork and communication.
- **Comprehensive Design Tools:** Figma provides a wide range of tools for crafting high-fidelity user interfaces and interactive prototypes. These include features for drawing shapes, creating vector graphics, adding text, applying colors and gradients, managing layers, and building prototypes with interactive links, animations, and transitions.
- **Developer Handoff:** Figma simplifies the developer handoff process by generating design specifications and assets directly from the designs. This ensures seamless translation of designs into code with accuracy and consistency between design and development teams.

6.1.2: Android Studio

Android Studio, developed by Google, is the official Integrated Development Environment (IDE) for Android app development. It offers a comprehensive suite of tools and features designed to streamline the process of designing, coding, debugging, and deploying Android applications. Below are the key features of Android Studio:

- **Optimized for Android Development**: Purpose-built for Android app development, Android Studio provides tools and functionalities tailored specifically to the Android platform, ensuring an efficient and seamless development experience.
- **Built-In Android Emulator:** The IDE includes a robust emulator that allows developers to test their applications on virtual Android devices with various configurations and Android versions. This enables the simulation of real-world scenarios to ensure compatibility across multiple devices.
- **Comprehensive Development Tools:** Android Studio provides a comprehensive set of development tools to support every stage of the app development process. It includes a Layout

Editor for designing intuitive user interfaces, an Intelligent Code Editor with features like code completion and refactoring to enhance coding efficiency, and Performance Profilers for analyzing and optimizing app performance. Additionally, it offers Advanced Debugging Tools to identify and resolve issues effectively and Testing Frameworks for creating and executing unit and UI tests, ensuring the reliability and quality of the application.

- **Seamless Integration with Google Services:** Android Studio integrates effortlessly with Google services and APIs like Firebase, Google Maps, and Google Cloud Platform. This enables developers to add features like real-time data synchronization, cloud storage, and location-based services to their apps with ease.
- **Version Control Support:** Android Studio includes built-in support for version control systems such as Git. This allows developers to manage repositories, track changes, collaborate with team members, and integrate version control workflows directly into the development process.

6.1.3: Flutter

Flutter [19] is an open-source framework developed by Google for building natively compiled applications for mobile, web, and desktop platforms from a single codebase. It uses the Dart programming language and features a widget-based architecture, where everything in the app is a widget, allowing for highly customizable and responsive user interfaces.

Flutter is known for its hot reload feature, enabling developers to instantly see changes without restarting the app, speeding up the development process. It compiles directly to native code, ensuring high performance with smooth animations and fast load times.

The framework supports both Material Design (Android) and Cupertino (iOS) widgets, making apps look and feel native on both platforms. It also integrates well with tools like Firebase for backend services and supports a range of plugins for additional functionality. With its growing ecosystem, Flutter is the best suit for cross-platform app development.

6.1.4: Visual studio code (VS Code)

Visual Studio Code (VS Code) is a free, open-source code editor created by Microsoft. Known for its lightweight yet powerful performance, it offers features like syntax highlighting, IntelliSense for intelligent code completion, and built-in debugging support. VS Code also includes integrated Git control for version management and provides access to a vast marketplace of extensions to extend its

functionality. With support for a wide array of programming languages and a high degree of customization, VS Code has provided efficiently code, debug, and manage their projects.

6.1.5: Firebase

Firebase [20] is a comprehensive platform developed by Google that offers a suite of backend services for mobile and web app development. It simplifies the process of building and managing applications by providing a range of tools for authentication, real-time data storage, messaging, and more. With Firebase Authentication, developers can easily handle user sign-ins through various methods such as email/password, social logins, and anonymous authentication. The platform also includes Firestore, a NoSQL database that allows for real-time syncing of app data across devices, ensuring seamless performance even when offline. For similar real-time functionality, Firebase Realtime Database allows instant data updates between users and devices.

Firebase also offers Cloud Storage for storing user-generated content like images and videos, while Cloud Messaging lets developers send push notifications to engage users. With Firebase Analytics, developers gain valuable insights into user behavior, enabling data-driven decisions to improve app performance and user experience. Additionally, Firebase Hosting provides a fast and secure platform for hosting static and dynamic web applications. By offering these tightly integrated services, Firebase enables developers to focus on app development without the need to manage backend infrastructure, making it a popular choice for building mobile and web apps.

6.1.6: Git and GitHub

Git is a distributed version control system designed to track changes in code during software development. It enables developers to efficiently manage different versions of code, monitor changes to files, and collaborate on projects.

GitHub, on the other hand, is a web-based platform that hosts Git repositories and provides additional tools for collaboration. It allows developers to store their repositories online, making it easier to share code, collaborate on projects, and contribute to open-source initiatives. GitHub includes features such as issue tracking, pull requests, code reviews, and project management tools, making it a widely used platform for hosting and managing code repositories.

6.2: Deployment diagram

The deployment diagram represents the physical architecture and interactions between different nodes and artifacts that take part in the deployment of the CareNest application. In this configuration, there are three primary nodes: User Mobile, Firebase Server, and Cradle. The User Mobile node contains the CareNest App, which is the client-side application that the users interact with. It talks to the Firebase Server using REST APIs, hence carrying out the main functionality of authentication for users and also data management capabilities.

The **Firebase Server** acts as a cloud-based back end hosting all important artifacts, which consist of Firebase Authentication and Firebase API services. They assist in providing the user authentication functionality securely, with an interface on which the mobile app could tap into any available functionalities, like storing data or creating real-time database capabilities.

The third node, **Cradle**, consists of two artifacts: Sensor Firmware and Communication Module. The Cradle node manages sensor data and facilitates communication between the Firebase Server and connected devices. It uses Wi-Fi HTTP protocols to connect with the Firebase Server to enable smooth data transmission and device management.

Overall, below deployment diagram effectively captures the relationships between the nodes and their respective artifacts, showcasing how the CareNest application operates within this distributed architecture. The interactions among these components ensure that user requests are processed efficiently while maintaining robust communication pathways across the system.

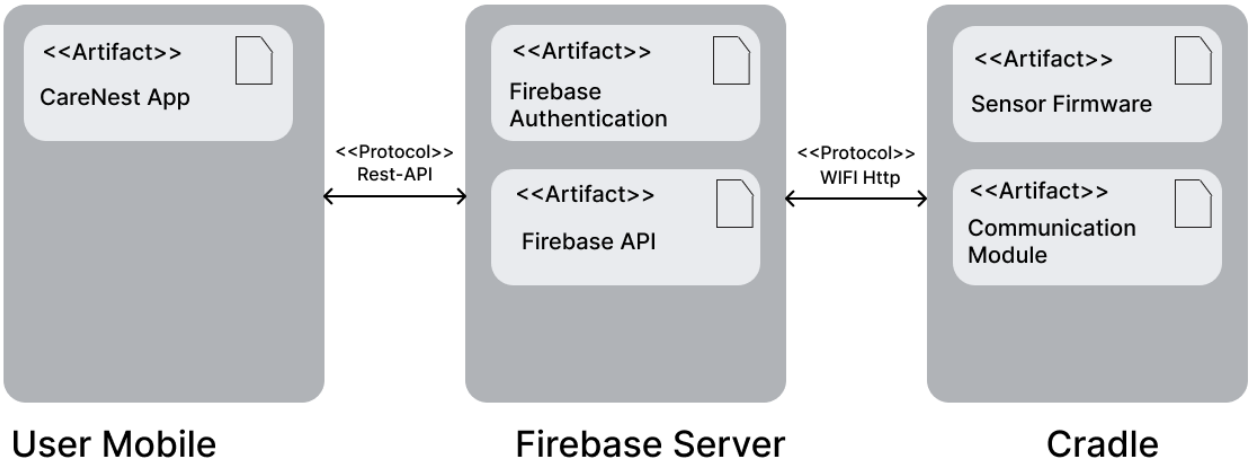


Figure 6. 1 Deployment Diagram of CareNest Application

6.3: Sequence diagram

A sequence diagram is a graphical representation of the sequence of interactions between objects or components of a system. It depicts how events occur in a sequence and how objects communicate with each other over time. It help in following: Model complex interactions: Visualize the sequence of events and interactions. Identify timing issues: Detect potential timing problems and optimize the sequence. Improve communication: Enhance understanding among team members and stakeholders.

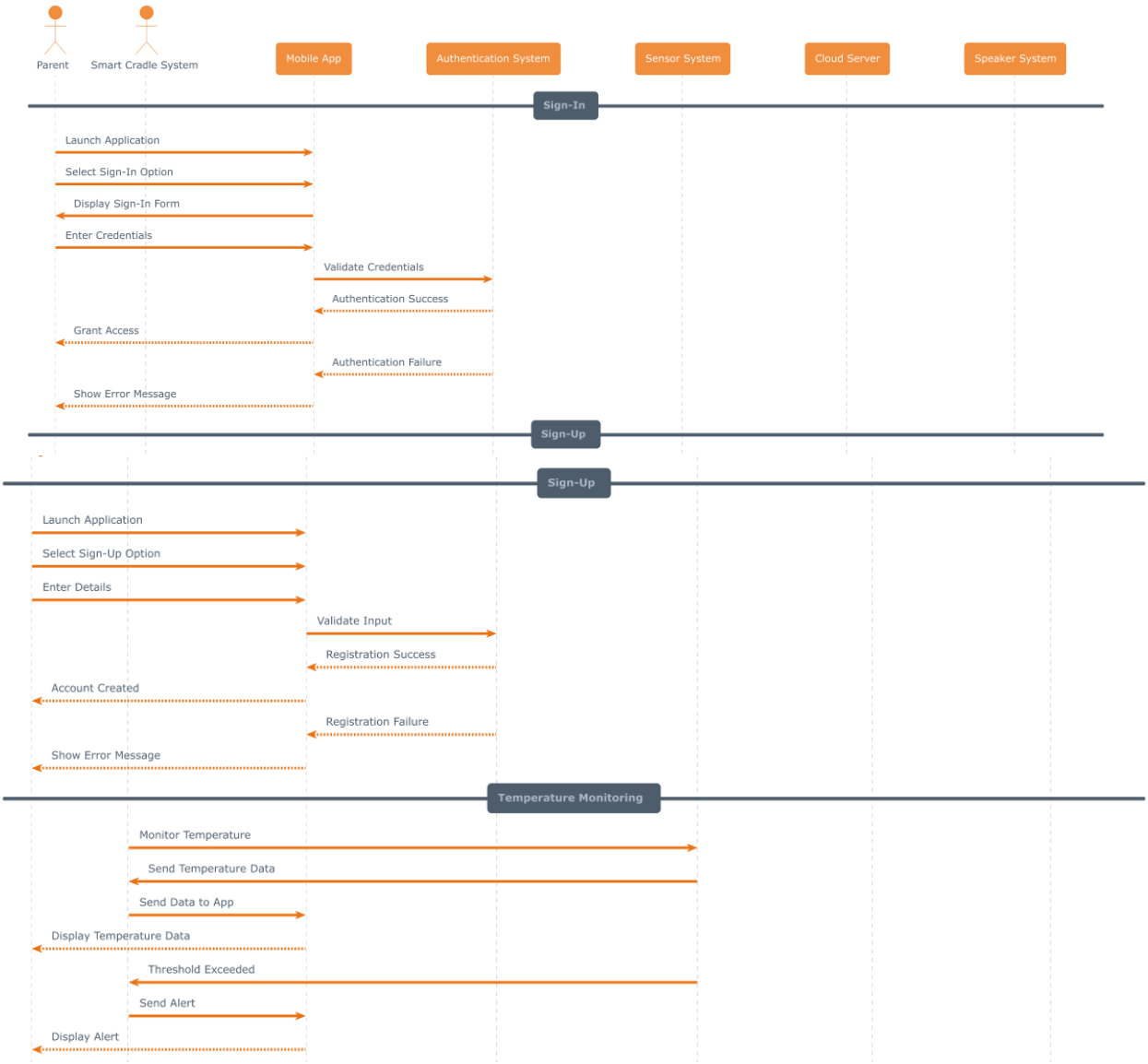
Components:

Participants: Objects or components involved in the interaction.

Lifelines: Vertical lines that represent the existence of the participants over time.

Messages: Horizontal arrows showing how participants interact.

Sequence: The sequence of events from top to bottom.


Parent
Smart Cradle System
Mobile App
Authentication System
Sensor System
Cloud Server
Speaker System
Sign-In
Launch Application
Select Sign-In Option
Display Sign-In Form
Enter Credentials
Validate Credentials
Authentication Success
Grant Access
Authentication Failure
Show Error Message
Sign-Up
Sign-Up
Launch Application
Select Sign-Up Option
Enter Details
Validate Input
Registration Success
Account Created
Registration Failure
Show Error Message
Temperature Monitoring
Monitor Temperature
Send Temperature Data
Send Data to App
Display Temperature Data
Threshold Exceeded
Send Alert
Display Alert

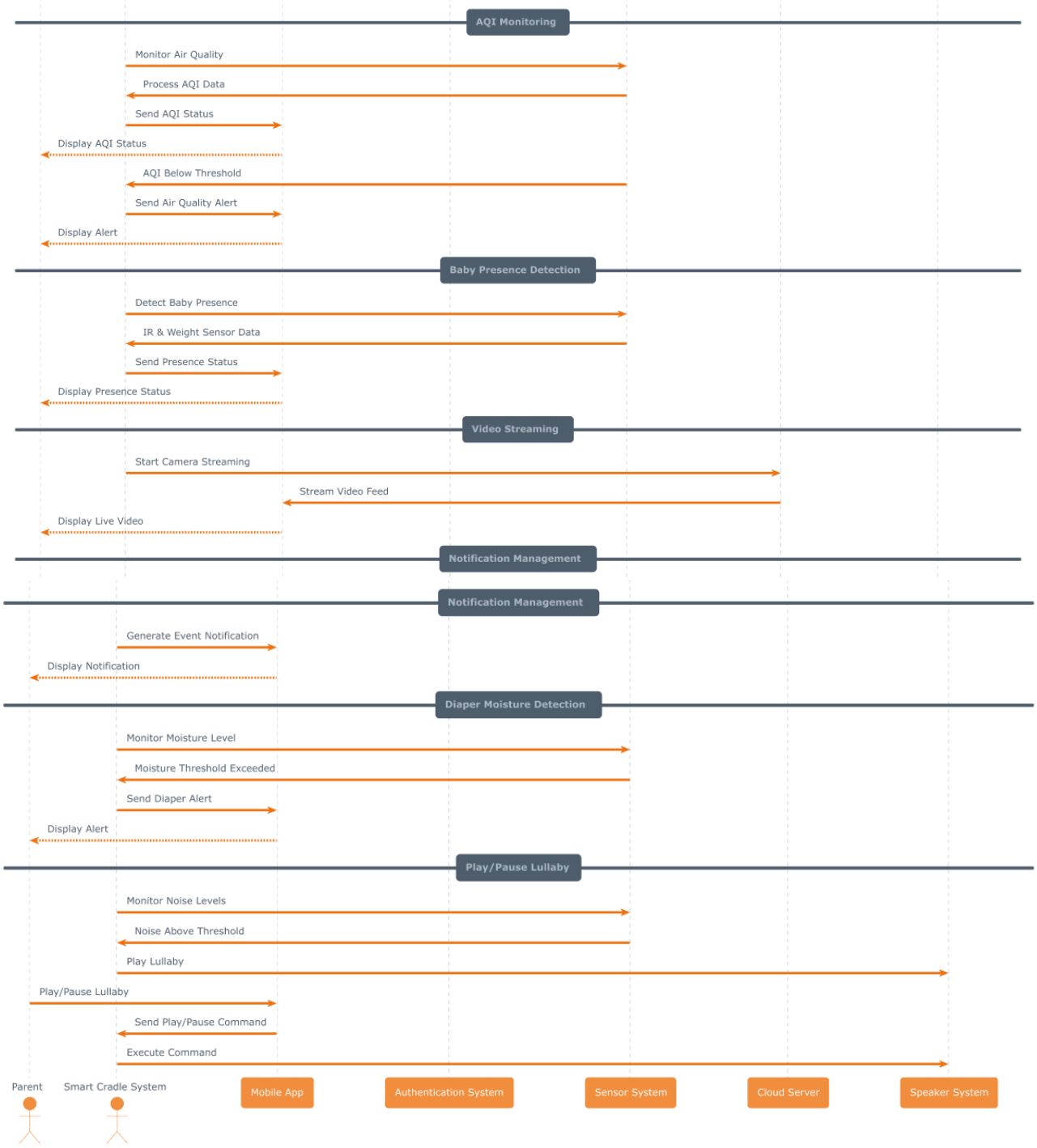


Figure 6. 2 Sequence Diagram of CareNest

6.4: NOSQL SCHEMA Diagram

It represents the visualization of a NoSQL database [21] structure and organization based on a NoSQL schema diagram. It visualizes the collections, documents, and fields along with their relationships to give a clear idea about the data model.

Collection: A NoSQL database equivalent of a relational database table; it is essentially a collection that stores related documents. Unlike the table, the collection does not enforce a fixed schema, thereby allowing the variety of fields in documents and differences in structure.

Document: A document is flexible, JSON-like data structure that contains in a NoSQL database: key-value pairs, arrays and nested objects. Every document acts as a record of a collection and can have any specific different structure of other documents which belong in the same collection.

Attributes/Properties: Also referred to as **fields**, are individual data elements that reside in a document. They are represented as attributes or properties.

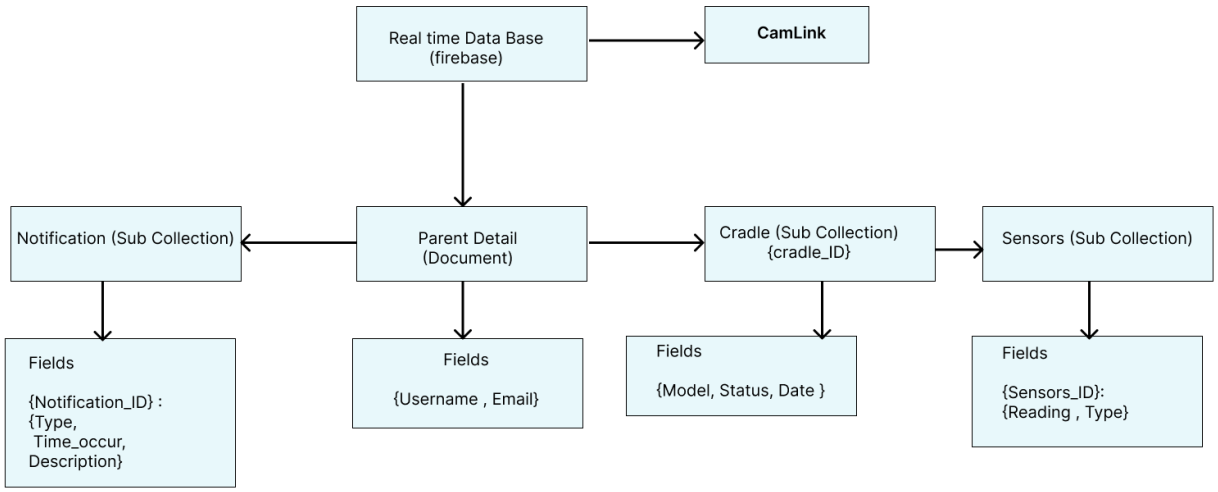


Figure 6. 3 NOSQL SCHEMA Diagram of Smart cradle monitoring system

Chapter 7: System Testing

7.1: System Testing

System Testing is a comprehensive testing process that validates the complete, integrated system to ensure it meets the specified requirements. Conducted after unit testing and integration testing, it focuses on evaluating the behavior of the system as a whole rather than its individual components. This type of testing is essential to confirm that all parts of the system work together seamlessly and deliver the intended functionality to users. The primary objectives of system testing include verifying functional requirements to ensure the system's features perform as specified, validating system integration to confirm that all modules and subsystems interact correctly, and identifying defects by detecting and addressing bugs or inconsistencies. It also involves evaluating non-functional requirements such as performance, reliability, scalability, and security, ensuring that the system meets user expectations. Additionally, system testing simulates real-world scenarios to verify system behavior under various conditions, including edge cases and exceptional situations. Key features of system testing include end-to-end testing, which evaluates the entire workflow of the system from input to output to ensure a seamless user journey.

The testing is conducted in an environment that closely resembles the production environment, allowing testers to identify and resolve issues before deployment. It also adopts a user-centric approach, focusing on ensuring the system delivers a high-quality user experience. System testing is critically important as it ensures the integrated system is free of defects, performs reliably, and is ready for deployment. By validating the system's functionality and performance, it provides confidence to stakeholders, developers, and end-users that the software meets their expectations and requirements.

7.1.1: Black Box Testing

Black box testing is an approach to software testing that involves testing the functionality of a system without considering its internal code, structure, or logic. This user-centric testing approach examines how the system behaves based on the inputs and outputs it has been given. Thus, this testing method would be ideal to ensure that the system behaves in a manner aligned with its requirements and user expectations. This approach to the system is done like an end-user, keeping their focus on verifying the external behavior without any knowledge of programming or internal implementation details.

Directly from requirements and specifications of the system comes test cases for black box testing, which gives strong alignment with what the users need. Some of the common techniques include equivalence

partitioning, which splits the inputs into valid and invalid classes to minimize redundant tests; boundary value analysis, where edge cases are analyzed because this is where errors are most likely to occur; decision table testing, used to test system behavior for input combinations; and state transition testing, ensuring the system behaves correctly during state changes.

This testing method has a number of advantages, such as simplicity, because it does not require any knowledge of the internal code of the system, and unbiased testing, since it focuses only on user interactions and outputs. Black box testing is widely used during system testing, acceptance testing, and regression testing to ensure that the system is ready for deployment, meets user expectations, and functions reliably under various conditions.

In a broader sense, the importance of black box testing is that it delivers a very high-level evaluation, user-oriented system, that would be intuitive and functional in terms of delivery of the correct requirement from the software product. So it proves its importance to give high-quality output while delivering to its business users by validating against requirements.

7.2: Test cases

7.2.1: Sign-in Test case

Table 7. 1 Successfully sign in using valid credentials

Field	Description
Test Case ID	1.1
Test Description	Verify that a parent can successfully sign in using valid credentials.
Preconditions	The parent has an existing account with valid credentials (email/username and password). The application is connected to the internet. The CareNest application is installed and functional.
Test Steps	1. Launch the CareNest application. 2. Select the "Sign In" option on the home screen. 3. Enter valid email/username and password in the respective fields. 4. Click the "Sign In" button.

	5. Wait for the system to validate the credentials
Expected Result	The system authenticates the user successfully. The user is redirected to the dashboard or main screen. The parent can access features like monitoring notifications, live video, and cradle statistics.
Actual Result	Pass
Status	Pass

Negative Test Cases:

Table 7. 2 Error message for invalid credentials.

Field	Description
Test Case ID	1.2
Test Description	Verify that the system displays an error message for invalid credentials.
Preconditions	The parent has an existing account with valid credentials (email/username and password). The application is connected to the internet. The CareNest application is installed and functional.
Test Steps	1. Launch the CareNest application. 2. Select the "Sign In" option. 3. Enter invalid email/username or password. 4. Click the "Sign In" button.
Expected Result	The system displays an error message: "Invalid email/username or password." The parent remains on the Sign In screen.
Actual Result	Pass
Status	Pass

Table 7. 3 Error message when there is no internet connection.

Field	Description

Test Case ID	1.3
Test Description	Verify that the system displays an error message when there is no internet connection.
Preconditions	The parent has valid credentials. The internet connection is disabled
Test Steps	1. Launch the CareNest application. 2. Select the "Sign In" option. 3. Enter valid email/username and password. 4. Click the "Sign In" button
Expected Result	The system displays an error message: "No internet connection. Please check your connection and try again." The parent remains on the Sign In screen.
Actual Result	Pass
Status	Pass

7.2.2: Sign-up Test case

Table 7. 4 Successfully sign up for an account using valid details

Field	Description
Test Case ID	2.1
Test Description	Verify that a parent can successfully sign up for an account using valid details.
Preconditions	The CareNest application is installed and functional. The application is connected to the internet.
Test Steps	1. Open the CareNest application. 2. Select the "Sign-Up" option on the home screen. 3. Enter the following valid details: - User Name - Valid Email Address - Password 4. Click the "Sign-Up" button. 5. Wait for the system to validate and register the account.

Expected Result	The system successfully registers the parent. The parent receives a confirmation message (e.g., "Account created successfully"). The parent can now log in using the registered credentials.
Actual Result	Pass
Status	Pass

Negative Test Cases:

Table 7. 5 Error message for an invalid email format

Field	Description
Test Case ID	2.2
Test Description	Verify that the system displays an error message for an invalid email format during sign-up.
Preconditions	The CareNest application is installed and functional. The application is connected to the internet.
Test Steps	1. Open the CareNest application. 2. Select the "Sign-Up" option. 3. Enter the following details: - Valid Full Name - Invalid Email Address (e.g., "user@com" or "user.com") - Valid Password. 4. Click the "Sign-Up" button.
Expected Result	The system displays an error message: "Invalid email format. Please enter a valid email address." The user remains on the Sign-Up screen.
Actual Result	Pass
Status	Pass

Table 7. 6 Error message when there is no internet connection

Field	Description

Test Case ID	2.3
Test Description	Verify that the system displays an error message when there is no internet connection during sign-up.
Preconditions	The application is installed but not connected to the internet.
Test Steps	1. Open the CareNest application. 2. Select the "Sign-Up" option. 3. Enter valid details for all required fields. 4. Click the "Sign-Up" button.
Expected Result	The system displays an error message: "No internet connection. Please check your connection and try again." The user remains on the Sign-Up screen.
Actual Result	Pass
Status	Pass

7.2.3: Temperature Monitoring Test case

Table 7. 7 sends alerts to the mobile app when thresholds are exceeded

Field	Description
Test Case ID	3.1
Test Description	Verify that the system monitors the baby's temperature and sends alerts to the mobile app when thresholds are exceeded.
Preconditions	The cradle is powered on. The temperature sensors (IR and Probe) are calibrated and functional. The mobile application is connected to the system via the IoT network.
Test Steps	1. Power on the smart cradle. 2. Place a baby in the cradle. 3. Ensure the Ultrasonic sensor detects the baby's presence. 4. Simulate the baby's body temperature within the normal range. 5. Gradually increase the temperature to exceed the upper threshold. 6. Observe the alert generated and data sent to the mobile app.

	7. Gradually decrease the temperature below the lower threshold and observe the behavior.
Expected Result	The system continuously monitors the baby's temperature. An alert is sent to the mobile app when the temperature exceeds or falls below the thresholds. The real-time temperature data is displayed correctly on the app.
Actual Result	Pass
Status	Pass

Negative Test Cases:

Table 7. 8 handles Temperature sensor malfunction

Field	Description
Test Case ID	3.2
Test Description	Verify that the system handles sensor malfunction gracefully and notifies the user.
Preconditions	The cradle is powered on. The temperature sensors (IR and Probe) are calibrated and functional. The mobile application is connected to the system via the IoT network.
Test Steps	1. Power on the smart cradle. 2. Simulate a sensor malfunction (e.g., disconnect the IR or Probe sensor). 3. Wait for the system to detect the malfunction. 4. Observe the notification or error message sent to the mobile app.
Expected Result	The system detects the sensor malfunction. An error message is sent to the mobile app (e.g., "Temperature sensor error detected").
Actual Result	Pass
Status	Pass

Table 7. 9 temperature sensor handles network failure

Field	Description

Test Case ID	3.3
Test Description	Verify that the system handles network failure gracefully and retries sending data.
Preconditions	The cradle is powered on. The temperature sensors (IR and Probe) are calibrated and functional. The mobile application is connected to the system via the IoT network.
Test Steps	1. Power on the smart cradle. 2. Place a baby in the cradle. 3. Disconnect the IoT network. 4. Monitor the system's behavior when sending temperature data. 5. Reconnect the network and observe if data transmission resumes.
Expected Result	The system detects the network failure. Alerts and data are queued for transmission. Upon reconnection, queued alerts and data are sent to the mobile app.
Actual Result	Pass
Status	Pass

Table 7. 10 Temperature sensor behavior when there is no baby in cradle

Field	Description
Test Case ID	3.4
Test Description	Verify that the system behaves correctly when no baby is detected in the cradle.
Preconditions	The cradle is powered on. The temperature sensors (IR and Probe) are calibrated and functional. The mobile application is connected to the system via the IoT network. the cradle is empty (no baby).
Test Steps	1. Power on the smart cradle. 2. Leave the cradle empty (no baby).

	3. Monitor the Ultrasonic sensor’s detection. 4. Observe the system’s behavior and notifications on the app.
Expected Result	The system detects that no baby is present. No temperature monitoring or alerts are triggered. A notification may be sent to the app (e.g., "Baby not detected in cradle").
Actual Result	Pass
Status	Pass

7.2.4: AQI Monitoring Test case

Table 7. 11 Send alerts when the AQI is poor

Field	Description
Test Case ID	4.1
Test Description	Verify that the system monitors the air quality and sends alerts to the mobile app when the AQI is poor.
Preconditions	The cradle is powered on. The air quality sensor is functional. The mobile application is connected to the system via the IoT network.
Test Steps	1. Power on the smart cradle. 2. Ensure the air quality sensor is functioning and measuring the air quality. 3. Simulate a poor air quality scenario by adjusting the environmental factors (e.g., increase pollution levels). 4. Wait for the AQI to be calculated and compared against the thresholds. 5. Verify that the system sends an alert for poor air quality. 6. Verify that the AQI status is updated and available for viewing on the mobile app.
Expected Result	The system calculates the AQI and provides a qualitative assessment (e.g., "Poor"). An alert is sent to the mobile app indicating unhealthy air quality. The AQI status is displayed in real-time on the app interface.
Actual Result	Pass

Status	Pass

Negative Test Cases:

Table 7. 12 Air quality sensor malfunction

Field	Description
Test Case ID	4.2
Test Description	Verify that the system handles air quality sensor malfunction and sends a notification to the user.
Preconditions	The cradle is powered on. The air quality sensor is functional. The mobile application is connected to the system via the IoT network.
Test Steps	1. Power on the smart cradle. 2. Simulate a malfunction in the air quality sensor (e.g., disconnect or disrupt the sensor). 3. Observe the system's behavior. 4. Wait for the system to detect the malfunction and send an error notification.
Expected Result	The system detects the malfunction in the air quality sensor. An error message is sent to the mobile app (e.g., "Air quality sensor malfunction detected"). The AQI status should not be updated in the app.
Actual Result	Pass
Status	Pass

Table 7. 13 AQI sensor when network failure

Field	Description
Test Case ID	4.3
Test Description	Verify that the system handles network failure and retries sending AQI data and alerts.
Preconditions	The cradle is powered on. The air quality sensor is functional.

	The mobile application is connected to the system via the IoT network.
Test Steps	1. Power on the smart cradle. 2. Disconnect the IoT network. 3. Simulate a poor air quality scenario. 4. Verify the system behavior when the network is unavailable. 5. Reconnect the network and verify that the data and alert are successfully sent to the mobile app
Expected Result	The system detects the network failure. The AQI data and alert are queued for transmission. Once the network is re-established, the data and alert are sent to the mobile app.
Actual Result	Pass
Status	Pass

Table 7. 14 AQI sensor inaccurate data handling

Field	Description
Test Case ID	4.4
Test Description	Verify that the system behaves correctly when the air quality data is inaccurate.
Preconditions	The cradle is powered on. The air quality sensor is functional. The mobile application is connected to the system via the IoT network.
Test Steps	1. Power on the smart cradle. 2. Simulate inaccurate air quality data (e.g., feeding incorrect sensor readings). 3. Observe the system’s response. 4. Verify whether the system detects the inaccurate data and sends an error message to the mobile app.
Expected Result	The system detects inaccurate air quality data. An error message is sent to the mobile app (e.g., "Inaccurate air quality data detected"). The AQI status should not be updated in the app.
Actual Result	Pass

Status	Pass

7.2.5: Baby Presence Detection Test case

Table 7. 15 detects the baby's presence in the cradle

Field	Description
Test Case ID	5.1
Test Description	Verify that the system correctly detects the baby's presence in the cradle and activates monitoring features.
Preconditions	The cradle is powered on. Both the IR and weight sensors are calibrated and functioning. The mobile application is connected to the system via the IoT network.
Test Steps	1. Power on the smart cradle. 2. Place a baby in the cradle. 3. Ensure both IR and weight sensors detect the presence of the baby. 4. Verify that the system cross-verifies input from both sensors. 5. Confirm that the system marks the cradle as "occupied." 6. Check that monitoring features (temperature, air quality, moisture) are activated. 7. Verify that the mobile app receives a notification indicating the baby’s presence and displays the real-time status.
Expected Result	Both the IR and weight sensors confirm the baby’s presence. The system activates monitoring features. The mobile app shows the "occupied" status and updates with real-time data.
Actual Result	Pass
Status	Pass

Negative Test Cases:

Table 7. 16 Baby presence malfunction

Field	Description

Test Case ID	5.2
Test Description	Verify that the system handles sensor malfunction and notifies the user.
Preconditions	The cradle is powered on. Both the IR and weight sensors are calibrated and functioning. The mobile application is connected to the system via the IoT network.
Test Steps	1. Power on the smart cradle. 2. Simulate a malfunction in either the IR or weight sensor (e.g., disconnect or disrupt the sensor). 3. Place the baby in the cradle. 4. Verify if the system detects the malfunction and responds accordingly. 5. Check for the error message or notification sent to the mobile app.
Expected Result	The system detects the sensor malfunction. An error message is sent to the mobile app (e.g., "Sensor malfunction detected"). Monitoring features are not activated
Actual Result	Pass
Status	Pass

Table 7. 17 baby presence detection sensor network failure

Field	Description
Test Case ID	5.3
Test Description	Verify that the system handles network failure and retries sending the baby presence status.
Preconditions	The cradle is powered on. Both the IR and weight sensors are calibrated and functioning. The mobile application is connected to the system via the IoT network.
Test Steps	1. Power on the smart cradle. 2. Place a baby in the cradle. 3. Disconnect the IoT network. 4. Verify the system’s behavior when the mobile app cannot receive updates.

	5. Reconnect the network and verify that the presence status is successfully updated and sent to the app.
Expected Result	The system detects the network failure. The presence status is queued for transmission. Once the network is re-established, the status is successfully sent to the mobile app.
Actual Result	Pass
Status	Pass

Table 7. 18 false detection of the baby's presence

Field	Description
Test Case ID	5.4
Test Description	Verify that the system handles false detection of the baby's presence (inaccurate data).
Preconditions	The cradle is powered on. Both the IR and weight sensors are calibrated and functioning. The mobile application is connected to the system via the IoT network.
Test Steps	1. Power on the smart cradle. 2. Simulate false detection (e.g., place an object with a similar weight or heat signature as the baby in the cradle). 3. Verify if the system detects the false presence. 4. Check if the system sends a notification indicating incorrect detection.
Expected Result	The system detects the false presence. A notification is sent to the mobile app (e.g., "False detection: Baby not present"). Monitoring features are not activated.
Actual Result	Pass
Status	Pass

7.2.6: Video Stream Test case

Table 7. 19 streams real-time video

Field	Description
Test Case ID	6.1
Test Description	Verify that the system streams real-time video correctly to the mobile application when the baby is present in the cradle.
Preconditions	The cradle is powered on. The camera is installed, configured, and functioning correctly. The mobile application is connected to the system via the IoT network. The user has a stable internet connection.
Test Steps	1. Power on the smart cradle. 2. Ensure the camera is operational and detects the baby's presence. 3. Wait for the system to start streaming the video feed. 4. Access the mobile application to view the video feed. 5. Confirm that the video feed is live and streaming in real-time. 6. Verify that the video stream continues as long as the baby is present, and the system remains active.
Expected Result	The camera streams the video feed to the cloud. The video feed is sent from the cloud to the mobile app. The user can view the live video feed on the mobile application without interruptions.
Actual Result	Pass
Status	Pass

Negative Test Cases:

Table 7. 20 : camera malfunction

Field	Description
Test Case ID	6.2
Test Description	Verify that the system handles camera malfunction and notifies the user.
Preconditions	The cradle is powered on. The camera is installed, configured, and functioning correctly.

	The mobile application is connected to the system via the IoT network. The user has a stable internet connection.
Test Steps	1. Power on the smart cradle. 2. Simulate a malfunction in the camera (e.g., disconnect or disrupt the camera feed). 3. Verify if the system detects the malfunction. 4. Check if the system sends a notification to the mobile app.
Expected Result	The system detects the camera malfunction. A notification is sent to the mobile app (e.g., "Camera malfunction detected"). The video stream is not available on the app.
Actual Result	Pass
Status	Pass

Table 7. 21 Camera sensor network failure

Field	Description
Test Case ID	6.3
Test Description	Verify that the system handles network failure and retries the video stream.
Preconditions	The cradle is powered on. The camera is installed, configured, and functions correctly. The mobile application is connected to the system via the IoT network. The user has a stable internet connection.
Test Steps	1. Power on the smart cradle. 2. Ensure the camera starts streaming. 3. Disconnect the IoT network. 4. Verify the system’s behavior when the mobile app cannot receive the video stream. 5. Reconnect the network and check if the video stream is resumed on the mobile app.
Expected Result	The system detects the network failure. The video stream is queued for retransmission. Once the network is re-established, the video stream is resumed.
Actual Result	Pass

Status	Pass

Table 7. 22 Camera sensor absence of the baby

Field	Description
Test Case ID	6.4
Test Description	Verify that the system handles the absence of the baby in the cradle and stops streaming.
Preconditions	The cradle is powered on. The camera is installed, configured, and functioning correctly. The mobile application is connected to the system via the IoT network. The user has a stable internet connection.
Test Steps	1. Power on the smart cradle. 2. Start the video streaming. 3. Remove the baby from the cradle. 4. Verify if the system detects that the baby is no longer in the cradle. 5. Check if the video feed is automatically stopped or paused.
Expected Result	The system detects that the baby is no longer present. The video stream stops or pauses. The mobile app does not display the live video feed once the baby is absent.
Actual Result	Pass
Status	Pass

7.2.7: Notification management Test case

Table 7. 23 delivers real-time notifications

Field	Description
Test Case ID	7.1

Test Description	Verify that the system generates and delivers real-time notifications for events such as temperature changes, air quality issues, diaper moisture alerts, and baby presence status.
Preconditions	The cradle system and mobile application are powered on and connected via the IoT network. All sensors and monitoring features are functioning correctly. Notifications are enabled in the mobile application settings.
Test Steps	1. Power on the cradle system and mobile app. 2. Enable notifications in the app settings. 3. Simulate an event (e.g., a temperature change, air quality issue, or diaper moisture alert). 4. Verify that the system generates the relevant notification. 5. Confirm that the notification is formatted correctly and includes necessary details (e.g., time, event type). 6. Verify that the notification is delivered to the mobile app in real-time. 7. Access the mobile app and confirm the notification is displayed. 8. Check that the user can take appropriate action based on the notification.
Expected Result	The system generates the notification in response to the event. The notification includes all relevant details (time, event type). The notification is delivered to the mobile app in real-time. The user is able to view and take action based on the notification
Actual Result	Pass
Status	Pass

Negative Test Cases:

Table 7. 24 Notification sensor malfunction

Field	Description
Test Case ID	7.2
Test Description	Verify that the system handles sensor malfunction and notifies the user correctly.

Preconditions	The cradle system and mobile application are powered on and connected via the IoT network. All sensors and monitoring features are functioning correctly. Notifications are enabled in the mobile application settings.
Test Steps	1. Power on the cradle system and mobile app. 2. Simulate a malfunction in one or more sensors (e.g., temperature sensor failure). 3. Verify that the system generates an error notification. 4. Check if the notification includes an error message (e.g., "Sensor malfunction"). 5. Verify that the notification is sent to the mobile app and displayed correctly.
Expected Result	The system detects the sensor malfunction. The system generates an error notification. The notification includes a relevant error message (e.g., "Sensor malfunction"). The notification is delivered to the mobile app.
Actual Result	Pass
Status	Pass

Table 7. 25 handles network failure

Field	Description
Test Case ID	7.3
Test Description	Verify that the system handles network failure and retries sending notifications.
Preconditions	The cradle system and mobile application are powered on and connected via the IoT network. All sensors and monitoring features are functioning correctly. Notifications are enabled in the mobile application settings.
Test Steps	1. Power on the cradle system and mobile app. 2. Simulate a network failure (disconnect the IoT network). 3. Trigger an event (e.g., temperature change). 4. Verify that the system detects the network failure and queues the notification for retransmission. 5. Reconnect the network and verify that the notification is sent to the mobile app.

Expected Result	The system detects the network failure. The notification is queued for retransmission. Once the network is re-established, the notification is delivered to the mobile app.
Actual Result	Pass
Status	Pass

Table 7. 26 : handles notifications when the baby is not detected

Field	Description
Test Case ID	7.4
Test Description	Verify that the system delivers notifications when the baby is not detected in the cradle.
Preconditions	The cradle system and mobile application are powered on and connected via the IoT network. All sensors and monitoring features are functioning correctly. Notifications are enabled in the mobile application settings.
Test Steps	1. Power on the cradle system and mobile app. 2. Ensure that the baby is not in the cradle. 3. Simulate an event (e.g., temperature alert, air quality issue). 4. Verify that the system generates a notification regarding the detected event. 5. Ensure that the notification is delivered to the mobile app.
Expected Result	The system generates the event notification even if the baby is not present. The notification is delivered to the mobile app.
Actual Result	Pass
Status	Pass

7.2.8: Diaper Moisture Detection Test case

Table 7. 27 detects moisture levels in the baby's diaper

Field	Description
Test Case ID	8.1

Test Description	Verify that the system detects moisture levels in the baby's diaper and sends an alert when a diaper change is required.
Preconditions	The cradle system is powered on. The probe moisture sensor is installed, calibrated, and functioning correctly. The mobile application is connected to the system via the IoT network.
Test Steps	1. Power on the cradle system and mobile app. 2. Ensure the moisture sensor is monitoring the diaper's moisture levels. 3. Simulate an event where the moisture level exceeds the pre-defined threshold. 4. Verify that the system detects the moisture level and generates an alert for diaper change. 5. Verify that the system sends the diaper change alert to the mobile app in real-time. 6. Confirm that the alert includes the necessary details (e.g., diaper change required). 7. Ensure the notification is displayed in the mobile app and is actionable.
Expected Result	The system accurately detects the moisture level exceeding the threshold. The system generates a timely alert. The alert is delivered to the mobile app in real-time. The user can view and take appropriate action based on the notification.
Actual Result	Pass
Status	Pass

Negative Test Cases:

Table 7. 28 Moisture sensor malfunction

Field	Description
Test Case ID	8.2
Test Description	Verify that the system handles sensor malfunction and generates a relevant error notification.

Preconditions	The cradle system is powered on. The probe moisture sensor is installed, calibrated, and functioning correctly. The mobile application is connected to the system via the IoT network.
Test Steps	1. Power on the cradle system and mobile app. 2. Simulate a malfunction in the probe moisture sensor (e.g., disconnect or cause failure). 3. Trigger an event where the system should detect moisture (even though the sensor is malfunctioning). 4. Verify that the system detects the malfunction and generates an error notification. 5. Check if the notification includes the error message (e.g., "Sensor malfunction"). 6. Confirm that the error notification is sent to the mobile app.
Expected Result	The system detects the sensor malfunction. The system generates an error notification with relevant details. The notification is delivered to the mobile app.
Actual Result	Pass
Status	Pass

Table 7. 29 Moisture sensor network failure

Field	Description
Test Case ID	8.3
Test Description	Verify that the system handles network failure while sending the diaper moisture alert.
Preconditions	The cradle system is powered on. The probe moisture sensor is installed, calibrated, and functioning correctly. The mobile application is connected to the system via the IoT network.
Test Steps	1. Power on the cradle system and mobile app. 2. Ensure that the moisture sensor detects a high moisture level and triggers a diaper change event. 3. Simulate a network failure (disconnect the IoT network). 4. Verify that the system detects the failure and queues the alert

	for retransmission. 5. Reconnect the network and verify that the alert is successfully sent to the mobile app.
Expected Result	The system detects the network failure. The diaper change alert is queued for retransmission. Once the network is re-established, the alert is delivered to the mobile app.
Actual Result	Pass
Status	Pass

Table 7. 30 generate false alerts

Field	Description
Test Case ID	8.4
Test Description	Verify that the system does not generate false alerts when moisture levels are within acceptable thresholds.
Preconditions	The cradle system is powered on. The probe moisture sensor is installed, calibrated, and functioning correctly. The mobile application is connected to the system via the IoT network.
Test Steps	1. Power on the cradle system and mobile app. 2. Ensure that the moisture sensor detects a moisture level that is below the threshold. 3. Simulate the event where the moisture level is within the acceptable range. 4. Verify that the system does not generate an alert when the moisture level is below the threshold. 5. Ensure that no unnecessary notifications are sent to the mobile app.
Expected Result	The system accurately detects that the moisture level is within the acceptable threshold. No false alerts are generated. No unnecessary notifications are sent to the mobile app.
Actual Result	Pass
Status	Pass

7.2.9: Play / Pause Lullaby Test case

Table 7. 31 plays or pauses the lullaby

Field	Description
Test Case ID	9.1
Test Description	Verify that the system plays or pauses the lullaby based on environmental noise or user input.
Preconditions	The cradle system is powered on. The sound sensor and speaker are installed and functioning correctly. The mobile application is connected to the system via the IoT network. Lullaby files are available on the speaker's SD card.
Test Steps	1. Power on the cradle system and mobile app. 2. Ensure the sound sensor is monitoring environmental noise. 3. Simulate noise exceeding the pre-defined threshold. 4. Verify that the system automatically plays a lullaby when the noise threshold is exceeded. 5. Verify that the lullaby is played through the speaker. 6. Manually pause the lullaby via the mobile app. 7. Verify that the lullaby stops playing. 8. Manually resume playing the lullaby via the mobile app. 9. Verify that the lullaby resumes from where it left off or starts a new track, based on the system behavior. 10. Verify that the lullaby stops automatically after the pre-set duration.
Expected Result	The system plays the lullaby automatically when the noise exceeds the threshold. The lullaby plays correctly through the speaker. The user can manually pause and resume the lullaby via the mobile app. The lullaby stops automatically after the pre-set duration
Actual Result	Pass
Status	Pass

Negative Test Cases:

Table 7. 32 sound sensor failure

Field	Description
Test Case ID	9.2
Test Description	Verify that the system handles sound sensor failure and does not play lullaby automatically.
Preconditions	The cradle system is powered on. The sound sensor and speaker are installed and functioning correctly. The mobile application is connected to the system via the IoT network. Lullaby files are available on the speaker's SD card.
Test Steps	1. Power on the cradle system and mobile app. 2. Simulate a failure in the sound sensor (e.g., disconnect it). 3. Simulate a noisy environment that should trigger the lullaby. 4. Verify that the lullaby does not play automatically due to the sound sensor failure. 5. Verify that an error notification is generated in the mobile app.
Expected Result	The system detects the sound sensor failure. The lullaby does not play automatically. An error notification is sent to the mobile app indicating the sound sensor failure.
Actual Result	Pass
Status	Pass

Table 7. 33 handles SD card issues

Field	Description
Test Case ID	9.3
Test Description	Verify that the system handles SD card issues and does not play a lullaby when the SD card is not accessible.
Preconditions	The cradle system is powered on. The sound sensor and speaker are installed and functioning correctly. The mobile application is connected to the system via the IoT

	network. Lullaby files are available on the speaker’s SD card.
Test Steps	1. Power on the cradle system and mobile app. 2. Simulate an issue with the SD card (e.g., unmount the card or corrupt the files). 3. Try to play the lullaby manually via the mobile app. 4. Verify that the system does not play the lullaby due to the SD card issue. 5. Verify that an error notification is sent to the mobile app indicating the SD card issue.
Expected Result	The system does not play the lullaby due to the SD card issue. An error notification is sent to the mobile app.
Actual Result	Pass
Status	Pass

Table 7. 34 lullaby network failure

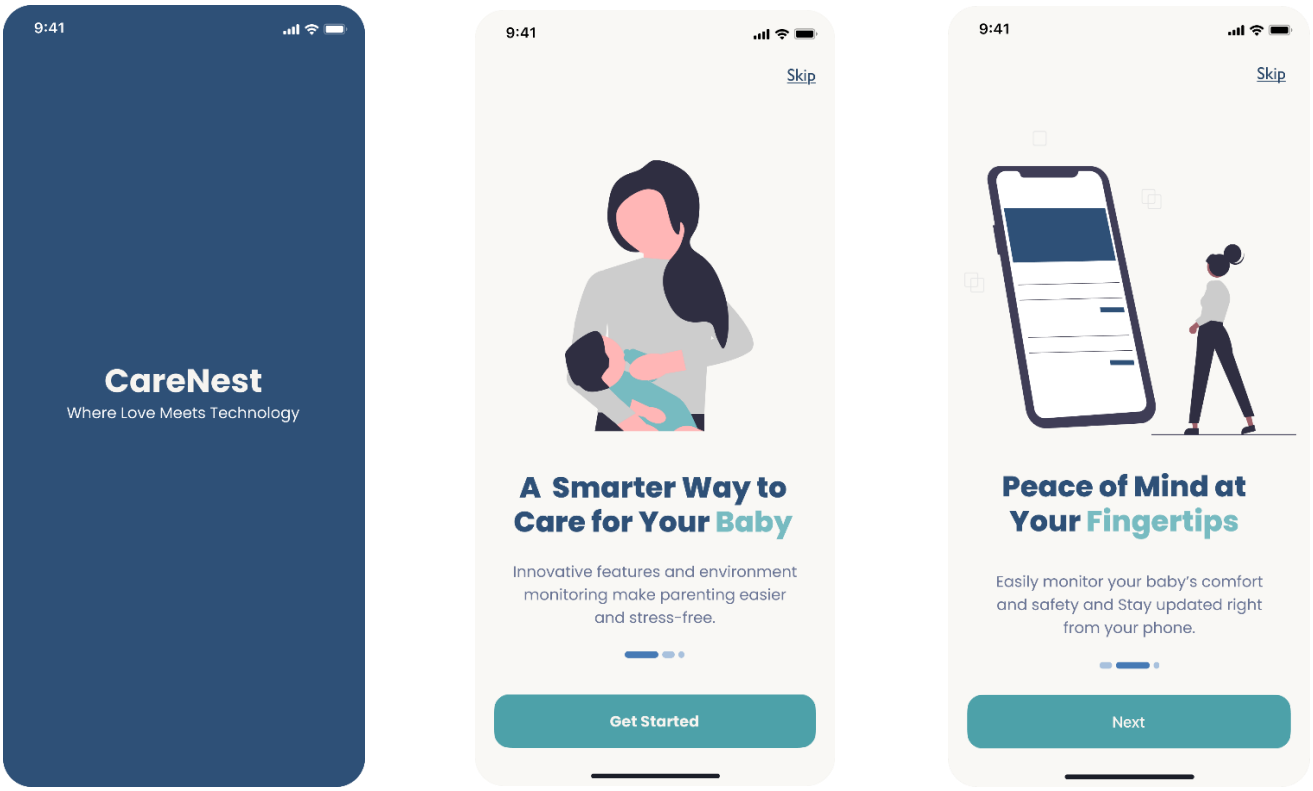
Field	Description
Test Case ID	9.4
Test Description	Verify that the system handles network failure while controlling the lullaby playback.
Preconditions	The cradle system is powered on. The sound sensor and speaker are installed and functioning correctly. The mobile application is connected to the system via the IoT network. Lullaby files are available on the speaker’s SD card.
Test Steps	1. Power on the cradle system and mobile app. 2. Play the lullaby manually via the mobile app. 3. Simulate a network failure (e.g., disconnect IoT network). 4. Verify that the lullaby playback is paused or interrupted due to network failure. 5. Reconnect the network and verify that the lullaby resumes or stops, as expected.
Expected Result	The system detects the network failure. The lullaby playback is paused or interrupted.

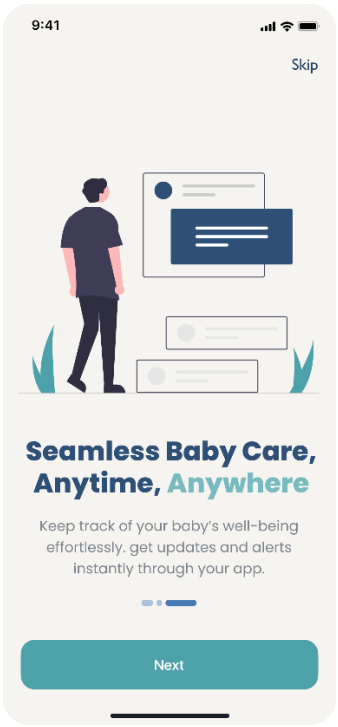
	The lullaby resumes or stops once the network is restored, as per the system behavior.
Actual Result	Pass
Status	Pass

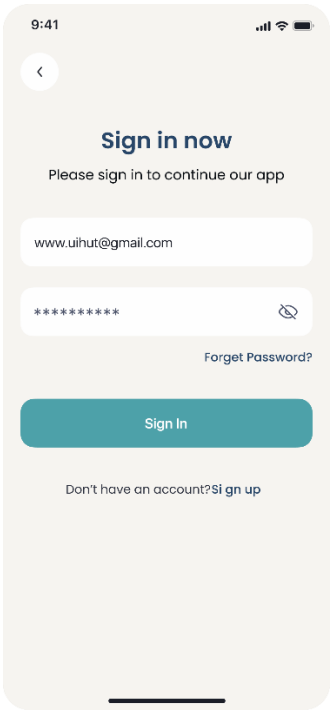
Chapter 8: Application Prototype

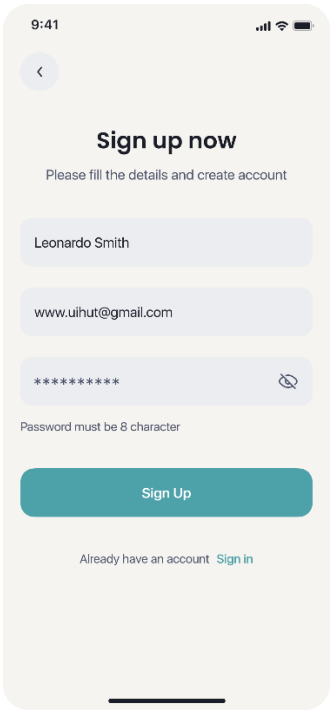
8.1: CareNest application prototype

Figure 8 1 CareNest application Screen Prototype Figure

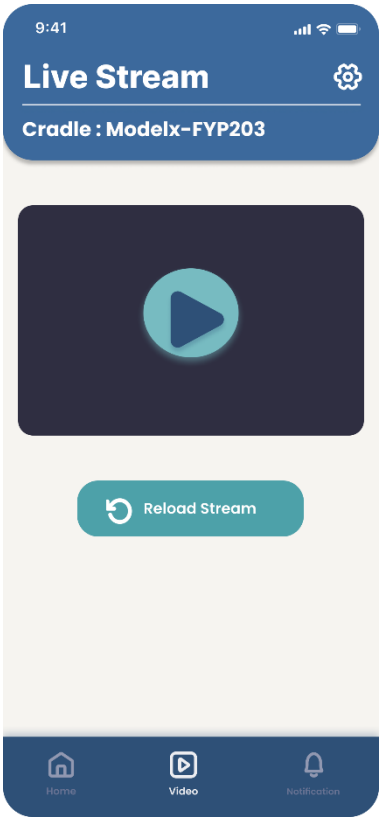


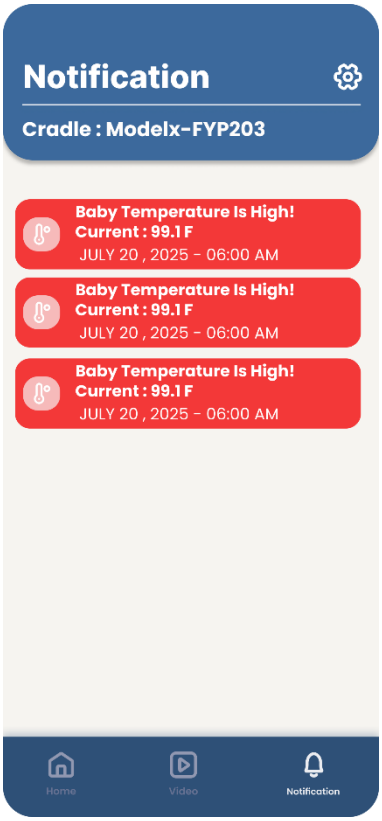

9:41
Skip
Seamless Baby Care,
Anytime, Anywhere
Keep track of your baby's well-being effortlessly. get updates and alerts instantly through your app.
Next

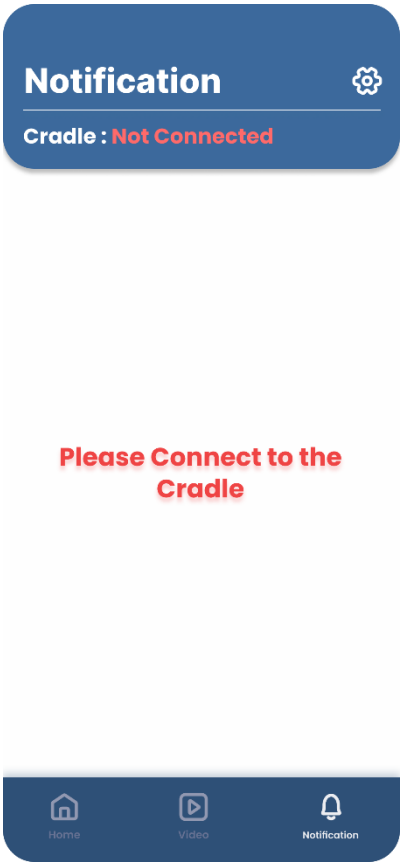

9:41
Sign in now
Please sign in to continue our app
www.uihut@gmail.com
****
Forget Password?
Sign In
Don't have an account?Si gn up

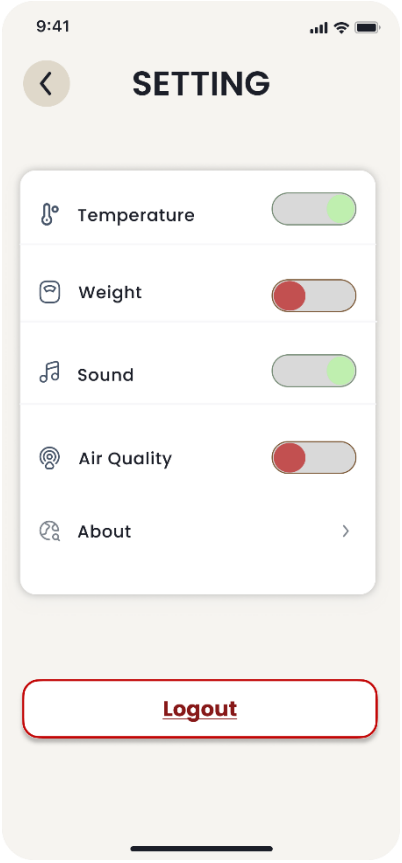

9:41
Sign up now
Please fill the details and create account
Leonardo Smith
www.uihut@gmail.com
****
Password must be 8 character
Sign Up
Already have an account Sign in

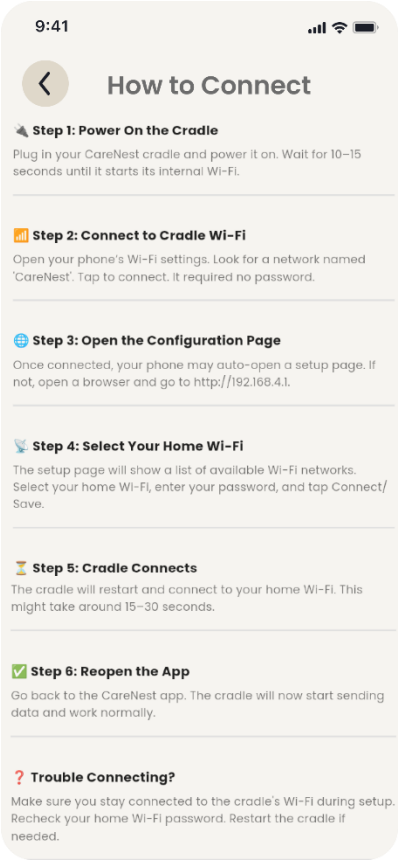

9:41
HOME
Cradle :Modelx-FYP203
Connect
Baby Presence : PRESENT
95.6°F
Baby body temperature
Diaper clean
Baby Diaper Moisture
Clam
Baby is crying or not
Low °
Air Quality
Home
Video
Notification


9:41
Live Stream
Cradle : Modelx-FYP203
Reload Stream
Home
Video
Notification


Notification
Cradle : Modelx-FYP203
Baby Temperature Is High!
Current : 99.1 F
JULY 20 , 2025 - 06:00 AM
Baby Temperature Is High!
Current : 99.1 F
JULY 20 , 2025 - 06:00 AM
Baby Temperature Is High!
Current : 99.1 F
JULY 20 , 2025 - 06:00 AM
Home
Video
Notification


Notification
Cradle : Not Connected
Please Connect to the Cradle
Home
Video
Notification


9:41
SETTING
Temperature
Weight
Sound
Air Quality
About
Logout


9:41
How to Connect
Step 1: Power On the Cradle
Plug in your CareNest cradle and power it on. Wait for 10–15 seconds until it starts its internal Wi-Fi.
Step 2: Connect to Cradle Wi-Fi
Open your phone's Wi-Fi settings. Look for a network named 'CareNest'. Tap to connect. It required no password.
Step 3: Open the Configuration Page
Once connected, your phone may auto-open a setup page. If not, open a browser and go to http://192.168.4.1.
Step 4: Select Your Home Wi-Fi
The setup page will show a list of available Wi-Fi networks. Select your home Wi-Fi, enter your password, and tap Connect/ Save.
Step 5: Cradle Connects
The cradle will restart and connect to your home Wi-Fi. This might take around 15–30 seconds.
Step 6: Reopen the App
Go back to the CareNest app. The cradle will now start sending data and work normally.
Trouble Connecting?
Make sure you stay connected to the cradle's Wi-Fi during setup. Recheck your home Wi-Fi password. Restart the cradle if needed.

References

[1] John. Hunt, “The unified process for practitioners : object-oriented design, UML and Java,” p. 280, 2000.

[2] “Newborn mortality.” Accessed: Jul. 07, 2025. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/newborn-mortality

[3] “World Health Organization (WHO).” Accessed: Jul. 07, 2025. [Online]. Available: https://www.who.int/

[4] “The Internet of Things for Health Care: A Comprehensive Survey | IEEE Journals & Magazine | IEEE Xplore.” Accessed: Jul. 07, 2025. [Online]. Available: https://ieeexplore.ieee.org/document/7113786

[5] “What is a Gantt Chart? Gantt Chart Software, Information, and History.” Accessed: Feb. 13, 2025. [Online]. Available: https://www.gantt.com/

[6] “Baby Monitor | Official VTech® Video Baby Monitors.” Accessed: Jul. 07, 2025. [Online]. Available: https://www.vtechbabycare.com/collections/video-monitors

[7] “Ellie A.I. Smart Baby Monitor Pro For Baby’s Sleep, Safety, Memories | Elliehello.com.” Accessed: Jul. 07, 2025. [Online]. Available: https://www.elliehello.com/

[8] “Fever - Symptoms & causes - Mayo Clinic.” Accessed: Jul. 07, 2025. [Online]. Available: https://www.mayoclinic.org/diseases-conditions/fever/symptoms-causes/syc-20352759

[9] “What is the Waterfall Model? Definition and Guide | Definition from TechTarget.” Accessed: Feb. 13, 2025. [Online]. Available: https://www.techtarget.com/searchsoftwarequality/definition/waterfall-model

[10] “What Is the V-Model? (Definition, Examples) | Built In.” Accessed: Feb. 13, 2025. [Online]. Available: https://builtin.com/software-engineering-perspectives/v-model

[11] C. Larman and V. R. Basili, “Iterative and Incremental Development: A Brief History,” *IEEE Computer*, vol. 36, no. 6, p. 47-56, 2003, doi: 10.1109/MC.2003.1204375.

[12] “Difference between Spiral model and Incremental model - GeeksforGeeks.” Accessed: Feb. 13, 2025. [Online]. Available: https://www.geeksforgeeks.org/difference-between-spiral-model-and-incremental-model/

[13] “What is Agile development?” Accessed: Feb. 13, 2025. [Online]. Available: https://indevlab.com/blog/what-is-agile-development/

[14] “SDLC-5: Rapid Application Development (RAD) Model - YouTube.” Accessed: Feb. 13, 2025. [Online]. Available: https://www.youtube.com/watch?v=29FFH0_lZyI

[15] “What Is Three-Tier Architecture? | IBM.” Accessed: Feb. 13, 2025. [Online]. Available: https://www.ibm.com/think/topics/three-tier-architecture

[16] “What is Unified Modeling Language (UML)?” Accessed: Feb. 13, 2025. [Online]. Available: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-uml/

[17] “RFC 7230 - Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing.” Accessed: Jul. 07, 2025. [Online]. Available: https://datatracker.ietf.org/doc/html/rfc7230

[18] Google, “Docs | Flutter.” Accessed: Feb. 13, 2025. [Online]. Available: https://docs.flutter.dev/

[19] Google, “Firebase Authentication.” Accessed: Feb. 13, 2025. [Online]. Available: https://firebase.google.com/docs/auth

[20] A. B. M. Moniruzzaman and S. A. Hossain, “NoSQL Database: New Era of Databases for Big data Analytics - Classification, Characteristics and Comparison,” Jun. 2013, Accessed: Feb. 13, 2025. [Online]. Available: http://arxiv.org/abs/1307.0191