PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA

MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH

YAHIA FARES UNIVERSITY OF MEDEA



Faculty of Technology

Department of Electrical Engineering

End of Study Project

Major: Electronics

Specialty: Electronics of Embedded Systems

Digital twin platform for neonatal incubator management

Presented by:

RAHMOUNI ikram.

LAYADI sidahmed.

Proposed and Supervised by:

Supervisor: Dr. BENYEZZA Hamza

Co-Supervisor: Prof. BOUHEDDA Mounir

June 2025

بسم الله الرحمن الرحيم

سبحانك لا علم لنا إلا
ما علمتنا إنك أنت
العليم الحكيم

صدق الله العظيم

Acknowledgments

First and foremost, we thank God for granting us the strength, courage, determination, and patience to complete this work.

We also extend our sincere thanks to Dr. Benyezza Hamza and Professor Dr. Bouhda Mounir, my supervisors, for their valuable guidance, continuous support, and constant encouragement throughout this journey. Their interest and patience were greatly appreciated.

We extend our sincere thanks to all members of the jury for their interest in our work.

We also wish to express our sincere thanks to the entire maintenance team at the Mohammed Boudiaf Hospital in Medea, the head of maintenance, Mr. Yekhlef Abdelhamid, and the head of neonatology, Mr. Ziani Yahia, who supported us with great patience and professionalism. and everyone who contributed, directly or indirectly, to the development of this project. We also extend our thanks to all our professors and faculty at Yahya Fares University in Medea.

Finally, we extend our deepest and most sincere gratitude to our dear parents and families for their sacrifices, assistance, support, and encouragement, which allowed us to pursue this Master's degree in the best possible conditio.

Dedication

All praise and thanks be to Allah, by whose grace good deeds are completed and wishes are fulfilled. I am deeply grateful for His guidance and blessings that accompanied me at every step, enabling me to reach this achievement.

To the ones whose prayers have always protected me,

To those who taught me patience, the value of effort, and the beauty of trust in God...

To my mother and father,

No success would ever be complete without dedicating it to you are the source, the light, and the blessing behind it all.

To my beloved brothers:

Reda, Hamza, Seif Eddine, Ghilas Abdelbasset

You have always been my silent strength, the joy behind my smile, and the unwavering support throughout this journey.

To my dear friends:

Asmaa, Akila, Fatma, Wafa, Ikram,

Your presence was a true gift your laughter, your advice, and your kindness meant the world to me during this journey.

You will always have a special place in my heart.

To my loyal friends:

Oussama, Abdelkader, and Taj Eddine

Thank you for being part of the path, for the support and encouragement that never went unnoticed.

And to my colleague and friend in this work:

Sid Ahmed

You were more than just a teammate you were a true partner throughout every step of this project.

Thank you for your commitment, your energy, and your trust.

I dedicate this achievement to you, with deep respect and appreciation.

Ikram.

Dedication

First and foremost, all praise and gratitude are due to Allah the Most Merciful who guided me through every challenge, lit my path with hope, and surrounded me with the most beautiful souls.

*To **my late father,***

may God have mercy on him, your memory is engraved in my heart. Not a day goes by without me wishing to share this moment with you. This success is a whisper of my eternal love for youa humble offering to your soul that continues to inspire me.

*To **my dear mother,***

may God protect and preserve her, your constant prayers were my shield, your love my compass, and your sacrifices my silent pillars of strength. Every page of this journey carries the pulse of your heart.

*To **my beloved siblings Hocine, Hamza, Amine, Samira, Fatima, and Anfal ,***

thank you for your quiet support, your love, and your faith in me.

*To **Yousra and little Celine, the cake of the house,***

you added warmth and joy to this journey.

*To **my faithful friends Abdelhak, Khaled, Alaeddine, and Abdelkrim***

your presence made the hard days easier.

*To **Cheikh Foudhil,***

your help meant more than words can say.

*To **Sidahmed Foudil,***

you are not just a friend you are a soul deeply cherished. Your kindness, sincerity, and quiet strength stayed with me in ways you'll never fully know. You were a light in my darkness.

*To **Zaki and Hassan,***

thank you for the laughter and sincere companionship.

*To **Ikram,***

my teammate and friend, your dedication and loyalty shaped this work more than you know.

*To **Dr. Hamza Benyeeza,***

your guidance guided me along the path ahead when I saw fog. You believed in me when I doubted myself your confidence was the wind that propelled me to this achievement.

*Finally, to **everyone** who stood by me seen or unseen thank you from the bottom of my heart. This achievement is ours.*

SidAhmed.

الملخص

يهدف مشروع التخرج هذا إلى إنشاء توأم رقمي لإدارة فعالة لحاضنات الأطفال حديثي الولادة. تم تثبيت لوحة إلكترونية (ESP32)متصلة بمجموعة من الحساسات لجمع بيانات بيئية وبيومترية مثل درجة الحرارة، الرطوبة، تركيز الأوكسجين، ومعدل ضربات القلب، وترسل البيانات في الوقت الحقيقي إلى سحابة .Firebase تعتمد المنصة أيضًا على نماذج LSTM للتنبؤ بالظروف البيئية وتفاعل الرضيع. تم تطوير واجهة ثلاثية الأبعاد تفاعلية باستخدام React و Blenderلعرض البيانات بطريقة مرئية سهلة للمستخدمين. تظهر نتائج الاختبار فعالية النظام المقترح في تحسين رعاية حديثي الولادة،

الكلمات المفتاحية :التوأم الرقمي؛ESP3 ؛Firebase ؛React ؛LSTM ؛ حاضنة الأطفال.

Abstract

This end of study project aims to create a digital twin for efficient neonatal incubator management. An ESP32 board is connected to environmental and biomedical sensors (temperature, humidity, oxygen, heart rate) to collect real-time data and sent it to Firebase. LSTM models are used to predict environmental conditions and infant comfort. An interactive 3D interface built with React and Blender allows intuitive data visualization. Test results demonstrate the proposed system's effectiveness in enhancing neonatal care, particularly in resource-limited healthcare settings.

Keywords: digital twin; ESP32; Firebase; React; LSTM; neonatal incubator.

Résumé

Ce projet de fin d'études vise à développer un jumeau numérique pour une gestion efficace des incubateurs néonataux. Une carte électronique (ESP32) est connectée à plusieurs capteurs pour collecter des données environnementales et biomédicales (température, humidité, oxygène, fréquence cardiaque), transmises en temps réel vers Firebase. Des modèles prédictifs LSTM sont utilisés pour anticiper les conditions critiques et le confort du nourrisson. Une interface interactive 3D, développée avec React et Blender, permet la visualisation intuitive des données. Les résultats des tests prouvent l'efficacité du système proposé dans l'amélioration des soins néonataux.

Mots clés : jumeau numérique ; ESP32 ; Firebase ; React ; LSTM ; incubateur néonatal.

Table of Contents

Acknowledgments I
Dedication II
Abstract IV
Table of Contents V
List of abbreviations IX
Liste of Figures XI
List of Tables XIV
General Introduction 1

Chapter I: Descriptive Study of the Incubator

I.1 Introduction to neonatal incubators 3
I.1.1 Definition and purpose 3
I.1.2 Historical evolution of incubators 3
I.1.3 Advancements and challenges in neonatal intensive care 3
I.2 Types of neonatal incubators 4
I.2.1 Closed-incubator systems 4
I.2.2 Open-incubator systems 5
I.2.3 Transport incubators 5
I.2.3 Hybrid incubators 6
I.2.4 Low-cost incubators 6
I.2.5 Neonatal intensive care unit (NICU) incubators 6
I.3 Key components and functionality 7
I.3.1 Temperature regulation (servo-control vs. manual) 7
I.3.2 Humidity and oxygen control 7
I.3.3 Air filtration and airflow mechanisms 7
I.3.4 Access ports and ergonomic design 8
I.4 Clinical applications and benefits 8
I.4.1 Premature infant care 8
I.4.2 Thermoregulation and infection prevention 9
I.4.3 Monitoring and safety features 10
I.5 Challenges and limitations 11
I.5.1 Noise and environmental stress 11
I.5.2 Maintenance and sterilization 11
I.5.3 Resource-limited settings limitations 12

I.6 Focus on the 8000SC closed incubator model 13
I.6.1 General presentation of the incubator 13
I.6.2 Operating principle 14
I.7 Conclusion 15

Chapter II: Overview of Digital Twin Technology

II.1 Introduction 16
II.2 Definition 16
II.2.1 Overview and functionality of digital twin 16
II.2.2. Digital model 17
II.2.3 Digital twin vs. digital model 18
II.3 Development and history of digital twin 19
II.4 Types of digital twin 21
II.4.1 Asset twin 21
II.4.2 System twins or unit twin 21
II.4.3 Process twin 21
II.4.4 Component twin 21
II.5 Components of a digital twin 21
II.5.1 Sensors 21
II.5.2 Data 22
II.5.3 Integration 22
II.5.4 Analytics 22
II.5.5 Actuators 22
II.6 Optimal applications and use cases for digital twins 22
II.6.1 Manufacturing 23
II.6.2 City Management 24
II.6.3 Aerospace 25
II.6.4 Healthcare 26
II.6.5 Agriculture 27
II.7 Key benefits of digital twins 27
II.8 Challenges 28
II.9 Digital twin: an integration of interconnected technologies 29
II.10 Different layers of the digital twin 30
II.10.1 Physical layer 30
II.10.2 Digital layer 30

II.10.3 Communication Layer 30
II.11 Digital twin platforms 30
II.11.1 AWS IoT twin maker 31
II.11.2 Azure digital twin 31
II.11.3 Eclipse ditto: 31
II.11.4 Siemens mindsphere 31
II.11.5 PTC thing worx 32
II.12 Conclusion 32

Chapter III: Software and Hardware Tools

III.1 Introduction 33
III.2 Software tools 33
III.2.1 Programming language – python 33
III.2.2 Visual studio code (VS Code) 33
III.2.3 Blender 37
III.2.4 Long short-term memory (LSTM) 40
III.2.5 Firebase 44
III.2.6 Firestore 46
III.3 Hardware tools 47
III.3.1 ESP-WROOM-32 47
III.3 Conclusion 48

Chapter IV : Design and Implementation

IV.1 Introduction 50
IV.2 System specifications 50
IV.3 Proposed system 50
IV.4 Implementation 51
IV.4.1 Physical layer 52
IV.4.2 Communication layer 53
IV.4.3 Cloud layer 54
IV.4.4 Prediction model layer 54
IV.4.4.1 3D Model of the incubator 54
IV.4.4.2 Predictive models 58
IV.4.4.3 Kalman filter 73

IV.4.4.4 Comparison of LSTM and Kalman 76
IV.4.5 Web application development 77
IV.4.5.1 Main interface 78
IV.4.5.2 Monitoring Interface 79
IV.4.5.3 History interface 81
IV.4.5.4 Control interface 81
IV.4.5.5 View3D interface 82
IV.4.5.6 Simulation interface 83
IV.4.5.7 Settings interface 84
IV.4.5.8 Charts interface 85
IV.5 Test results 85
IV.5.1 Validating data transfer and cloud storage 86
IV.5.2 Simulated data entry and Firestore storage validation 86
IV.5.3 Simulation test 88
IV.5.4 Visualization 3D incubator 88
IV.6 Conclusion 89
General Conclusion 90
References 91

List of abbreviations

2D: Two-Dimensional
3D: Three-Dimensional
3D CAD: 3D Computer-Aided Design
ADC: Analog to Digital Converter
AI: Artificial Intelligence
Android SDK: Android Software Development Kit
API: Application Programming Interface
Arduino IDE: Arduino Integrated Development Environment
ASP: Active Server Pages
AWS: Amazon Web Services
BLE: Bluetooth Low Energy
CPU: Central Processing Unit
CRM: Customer Relationship Management
CPS: Cyber-Physical Systems
DAC: Digital to Analog Converter
DT: Digital Twin
ERP: Enterprise Resource Planning
ESP32: Espressif Systems Wi-Fi and Bluetooth Microcontroller
ESP-IDF: Espressif IoT Development Framework
EV: Electric Vehicle
HMI: Human-Machine Interface
HTML: HyperText Markup Language
I2C: Inter-Integrated Circuit
IoT: Internet of Things
KMC: Kangaroo Mother Care
LSTM: Long Short-Term Memory
M2M: Machine to Machine
macOS: Macintosh Operating System
MCU: Microcontroller Unit
MES: Manufacturing Execution System
MP3: MPEG Audio Layer III
NASA: National Aeronautics and Space Administration
NET: Network Enabled Technology (commonly refers to Microsoft .NET Framework)
NICU: Neonatal Intensive Care Unit
NIDCAP: Newborn Individualized Developmental Care and Assessment Program
NoSQL: Not Only SQL
NSF: National Science Foundation
OTA: Over-The-Air (firmware or software updates)
PaaS: Platform as a Service
PlatformIO: Embedded Development Platform
PLM: Product Lifecycle Management
PTC: Parametric Technology Corporation
PWM: Pulse Width Modulation
R^2: Coefficient of Determination
RMSE: Root Mean Square Error

SCBUs: Special Care Baby Units
SD: Secure Digital
SDIO: Secure Digital Input Output
SPI: Serial Peripheral Interface
SRAM: Static Random-Access Memory
SSL: Secure Sockets Layer
the US: The United States
TLS: Transport Layer Security
UART: Universal Asynchronous Receiver-Transmitter
URL: Uniform Resource Locator
VLBW: Very Low Birth Weight
VS Code: Visual Studio Code
Wi-Fi: Wireless Fidelity

Liste of Figures

Chapter I: Descriptive Study of the Incubator

Figure I. 1: Neonatal intensive-care unit from 1980. 3
Figure I. 2: A premature infant, intubated and requiring mechanical ventilation. 4
Figure I. 3: Closed-Incubator. 4
Figure I. 4: Open -Incubator. 5
Figure I. 5: Transport Incubators. 5
Figure I. 6: Hybrid Incubators. 6
Figure I. 7: Low-Cost Incubators. 6
Figure I. 8: Neonatal Incubator Components and Functionality. 8
Figure I. 9:Steps to Enhance Premature Infant Care. 9
Figure I. 10: NICU Challenges and Solutions. 13
Figure I. 11: DRAGER “8000sc” incubator. 14
Figure I. 12: General diagram of operation of sensors and actuators. 15

Chapter II: Overview of Digital Twin Technology

Figure II. 1: Overview of the digital twin. 17
Figure II. 2: Evolution of digital twin. 21
Figure II. 3: Digital Twin in Maritime and Shipping. 23
Figure II. 4: digital twin in manufacturing. 24
Figure II. 5: Digital twin city management. 25
Figure II. 6: digital twin in aerospace. 25
Figure II. 7:Application and prospects of digital twin technology in the medical field. 26
Figure II. 8: The Connected Farmer: A Digital Twin Approach to Smart Agriculture. 27
Figure II. 9: Layers of the Digital Twin. 29

Chapter III: Software and Hardware Tools

Figure III. 1: VS code Official download page. 35
Figure III. 2: Software VS Code. 36
Figure III. 3: Creating a REACT project. 37
Figure III. 4: Initial Blender Interface. 38
Figure III. 5: Modeling Tools in Blender. 38
Figure III. 6: Material Creation in Blender. 38
Figure III. 7: Blender Animation Workflow. 39
Figure III. 8: Lighting Tools in Blender. 39
Figure III. 9: Python Scripting in Blender. 40
Figure III. 10: LSTM components. 41
Figure III. 11: LSTM Cell Stat. 41
Figure III. 12: LSTM Forget Gate. 42
Figure III. 13: LSTM Input gate. 42
Figure III. 14: LSTM Output gate. 43
Figure III. 15: Naming the Firebase Project. 44
Figure III. 16: Firebase Project Settings. 45

Figure III. 17: Realtime Database Creation Page. 45
Figure III. 18: Firebase Database URL. 46
Figure III. 19: Firestore Database Setup Page. 47
Figure III. 20: ESP-WROOM-32. 48

Chapter IV: Design and Implementation

Figure IV. 1: General architecture of the proposed system. 51
Figure IV. 2: System architecture overview. 52
Figure IV. 3: Physical Incubator. 52
Figure IV. 4: part of the arduino program. 53
Figure IV. 5: Configuring wi-Fi on Arduino. 53
Figure IV. 6: Connecting ESP32 to Firebase. 53
Figure IV. 7: Lower Base. 55
Figure IV. 8: Curved handles. 55
Figure IV. 9: Curved legs. 56
Figure IV. 10: Transparent dome. 56
Figure IV. 11: Symbolic element. 57
Figure IV. 12: Model of the Incubator. 57
Figure IV. 13: Part of the thermal dataset script. 58
Figure IV. 14: Part of the humidity dataset script. 59
Figure IV. 15: Part of the CO_2 dataset script. 59
Figure IV. 16: Part of the baby response dataset script. 60
Figure IV. 17: Baseline LSTM results. 63
Figure IV. 18: LSTM with Dropout results. 63
Figure IV. 19: LSTM + Dense results. 64
Figure IV. 20: Bidirectional LSTM results. 64
Figure IV. 21: Baseline LSTM results. 65
Figure IV. 22: LSTM with Dropout results. 66
Figure IV. 23: LSTM + Dense results. 66
Figure IV. 24: Bidirectional LSTM results. 67
Figure IV. 25: Baseline LSTM results. 68
Figure IV. 26: LSTM with Dropout results. 68
Figure IV. 27: LSTM + Dense results. 69
Figure IV. 28: Bidirectional LSTM results. 69
Figure IV. 29: Baseline LSTM results. 70
Figure IV. 30: LSTM + Dropout results. 71
Figure IV. 31: LSTM + Dense results. 71
Figure IV. 32: Bidirectional LSTM results. 71
Figure IV. 33: Kalman filter scripts, (1) temperature, (2) oxygen, (3) humidity, (4) baby-response. 74
Figure IV. 34: Kalman Filter Temperature. 74
Figure IV. 35: Kalman filter oxygen. 75
Figure IV. 36: Kalman filter humidity. 75
Figure IV. 37: Kalman filter baby response. 75
Figure IV. 38: Standard components. 78
Figure IV. 39: Main interface. 79

Figure IV. 40: Monitoring interface. 80
Figure IV. 42: History. 81
Figure IV. 43: Control interface. 82
Figure IV. 44: View3D Interface. 82
Figure IV. 45: Simulation interface (1). 83
Figure IV. 46: Simulation interface (2). 83
Figure IV. 47: Settings Interface. 84
Figure IV. 48: Drawings Interface. 85
Figure IV. 49: Information stored in the Realtime Database (1). 86
Figure IV. 50: Information stored in the Realtime Database (2). 86
Figure IV. 51: Information stored in the Firestore (1). 87
Figure IV. 52: Information stored in the Firestore (2). 87
Figure IV. 53: Information stored in the Firestore (3). 87
Figure IV. 54: Predictive results 88
Figure IV. 55: virtual incubator. 89

List of Tables

Chapter II: Overview of Digital Twin Technology

Table II. 2 the difference between digital twins and digital model 18

Chapter III: Software and Hardware Tools

Table III. 1 illustrates the features of the ESP-WROOM-32 module. 48

Chapter IV: Design and Implementation

Table IV. 1 LSTM Model performance for predicting incubator temperature 65
Table IV. 2 LSTM Model performance for predicting incubator humidity 67
Table IV. 3 LSTM Model performance for predicting incubator oxygen-concentration..... 69
Table IV. 4LSTM Model performance for predicting incubator baby-response 72

General Introduction

General Introduction

The neonatal period, defined as the first 28 days of life, is one of the most vulnerable periods of human development, especially for low birth weight or preterm babies. These newborns are at significant risk of morbidity and mortality due to physiological immaturity, especially with regard to thermoregulation, respiration, and immunity. It appears that 15 million babies are born prematurely every year, and over 1 million deaths are related to complications of preterm birth [1].

As a means of attempting to care for these vulnerable newborns, neonatal incubators have been a standard of practice since early development in the 19th century. From rudimentary heat boxes to present advanced machinery, incubators have evolved into regulated microenvironments that adjust temperature, humidity, oxygen concentration, and other essential parameters to facilitate survival and growth [2]. Despite technology, traditional incubators continue to be largely reactive rather than proactive; they lack the intelligent potential to foretell external occurrences before they occur.

Over the past few years, the convergence of new technologies such as the Internet of Things (IoT), cloud computing, and artificial intelligence (AI) has created new windows of opportunities for innovation in the healthcare sector. The most groundbreaking paradigm among them is the Digital Twin (DT). Initially thought about by Michael Grieves in 2003 and later utilized by NASA for mission-critical simulation [12][13], a digital twin is an interactive digital replica of a physical system in real time. It facilitates continuous synchronization between the real and virtual worlds to allow monitoring, simulation, and predictive analysis across the whole system lifecycle [5].

Our project, in this sense, proposes a DT platform for neonatal incubator management, which aims to enhance neonatal care quality through the application of smart automation, real-time analysis, and remote visualization. The platform is designed with a multilayer architecture that employs:

- An embedded system based on ESP32, integrated with environmental and biomedical sensor devices (temperature, humidity, oxygen, heart rate, etc.).
- A Firebase cloud platform for receiving and storing data in real time, highly available, and horizontally scalable.
- Predictive models using Long Short-Term Memory (LSTM) networks to forecast environmental conditions, as well as infant stress indicators.
- An interactive 3D visualization interface developed using React and Blender, providing healthcare professionals with an immersive, user-friendly experience.

The aim of this system is to close the loop between monitoring and intelligent decision-making. Through the transformation of static data into meaningful insights, the digital twin can facilitate timely intervention, prevent human error, and improve clinical results in Neonatal Intensive Care Units (NICUs).

Additionally, our system aligns with the Industry 4.0 and Healthcare 4.0 paradigms, which emphasize connectivity, automation, and individualization [6]. In low-resource settings, where neonatal mortality will be highest, such innovation has the potential to make an important

contribution towards maximizing scarce medical resources and accessing higher quality care [7].

This manuscript consists of four chapters: chapter one provides a descriptive introduction to neonatal incubators and their history; chapter two explores the roots and applications of DT technology; chapter three presents the hardware and software tools utilized during the implementation of the system; and chapter four discusses the design, implementation, and performance evaluation of the proposed platform.

Chapter I

Descriptive Study of the Incubator

I.1 Introduction to neonatal incubators

I.1.1 Definition and purpose

A neonatal intensive care unit (NICU) is an isolated unit within a hospital equipped with life-supporting machines and specialized care for preterm or sick newborns. There are three categories in which infants can be allocated in NICUs: critical care, intermediate care, and step-down units. NICUs were established in the United States in 1960 by Louis Gluck, who set the precedent for the role of a neonatologist. Interrupting maternal and child health care practices will always leave a gap. “Neonatal” describes an infant, less than 28 days old [1].

I.1.2 Historical evolution of incubators

Care practices focused on premature babies have been around since the 17th century, with more strides made towards the 1800s

- Designed by stephane trainer 1880: the first closed incubator, modeled from the designs of poultry incubators.
- Pierre Budin, amid the growing popularity of breastfeeding emphasized its importance, was midwife and protector of infant lactation.
- Enhanced design of the incubators in 1890, introduced by Alexandre Lion. It became well recognized under the courtesy of martin Cooney, who marketed these premature infants showcased in incubators.
- In 1931 A. robert bauer invented the incubator system to integrate warmth, oxygen, and humidity synergistically.
- Post-war period 1945 was the turning point for Special Care Baby Units (SCBUs) came up in Bristol and Birmingham, UK. The 1960s ushered major advancements, one of which was the construction of St Thomas’s tube for safe delivery of oxygen.
- By the 1970s–1980s, (NICUs) introduced the technology of centralized units and custom monitoring, which were fully integrated into hospital settings. Even so, as shown in figure I.1, the application of technology to areas such as mother-infant bonding was becoming an issue of growing concern.



Figure I. 1: Neonatal intensive-care unit from 1980.

I.1.3 Advancements and challenges in neonatal intensive care

Contemporary NICUs focus baby care facilities on extremely premature or ill newborns. The main efforts are directed to:

- Elevating the survival rates of children under 1 kilogram and even curtis means in 2020 after the birth at 21 weeks via surgical process.
- Involvement of parents great emphasis on skin-to-skin contact “kangaroo care” and encouragement of nonrestrictive environments [2].
- Reducing long-term complications such as: Cerebral paresis of accomplish, possibility the patient in some degree learning disability, and handicap of vision or hearing disable.
- Limiting the risks of the most severe complications such as breathing problems, infections, necrotizing enterocolitis, and brain hemorrhage is essential in neonatal care, as illustrated in figure I.2**,** A premature infant, intubated and requiring mechanical ventilation.



Figure I. 2: A premature infant, intubated and requiring mechanical ventilation.

I.2 Types of neonatal incubators

I.2.1 Closed-incubator systems

These are traditional incubators with a completely enclosed environment. They typically consist of a clear plastic box with access doors for medical staff to attend to the infant. Closed systems provide precise control over temperature, humidity, and oxygen levels (figure I.3) [11].



Figure I. 3: Closed-Incubator.

I.2.2 Open-incubator systems

Also known as radiant warmers or open-care systems, these incubators provide a warm environment for the infant without enclosing them in a completely sealed chamber. Instead, they use overhead heaters to maintain the desired temperature while allowing easier access to the baby for medical procedures and care, as seen in figure I.4.



Figure I. 4: Open -Incubator.

I.2.3 Transport incubators

These are portable units designed for transporting premature or critically ill infants between medical facilities. Transport incubators, as illustrated in figure I.5, are equipped with features like shock absorption, battery backup, and monitoring capabilities to ensure the safety and stability of the infant during transit.



Figure I. 5: Transport Incubators.

I.2.3 Hybrid incubators

These combine features of both closed and open systems, offering the flexibility of access provided by open systems while also maintaining the controlled environment of closed systems when necessary, as shown in figure I.6. Hybrid incubators often feature removable hoods or access ports for medical procedures.



Figure I. 6: Hybrid Incubators.

I.2.4 Low-cost incubators

In recent years, there has been a push to develop low-cost, portable, and easy-to-use infant incubators, particularly for use in developing countries where access to medical resources is limited. These incubators utilize simple designs and affordable materials while still providing essential temperature control and monitoring functions, as illustrated in figure I.7.



Figure I. 7: Low-Cost Incubators.

I.2.5 Neonatal intensive care unit (NICU) incubators

These are specialized incubators used in (NICUs) to provide intensive care for premature or critically ill newborns. NICU incubators may incorporate advanced features such as integrated monitoring systems, respiratory support equipment, and specialized beds for infants with specific medical needs.

Infant baby incubators play a crucial role in providing a supportive environment for the fragile health of newborns, particularly those who require extra care and monitoring in the early stages of life.

I.3 Key components and functionality

Neonatal incubators are sophisticated medical devices designed to establish an artificial, protective habitat for preterm or critically ill newborns. These incubators offer multiple integrated systems to provide optimal physiological stability through accurate management of temperature, humidity, oxygen and sterility. Here is a summary of the main components and what each one does:

I.3.1 Temperature regulation (servo-control vs. manual)

However, it is essential to preserve thermal balance within the neonatal period as preterm newborns have an innate immature thermoregulatory mechanism. There are two basic methods used by incubators to regulate temperature such as manual control and servo-control systems.

Manual control systems rely on healthcare personnel to set the air temperature following periodic clinical evaluations. Although they can be effective, they are prone to human error and can lead to temperature variations that are detrimental to the infant's state.

Servo-controlled incubators, in contrast, automatically controlled environmental conditions within the incubator by measuring skin temperature via sensors placed on the body of the infant. The sensor sends real-time data to a microcontroller which instantaneously regulates the heating elements of the incubator to maintain a constant skin temperature around the physiological ideal of ~36.5°C [10]. Servo-control is superior to the prevention of hypothermia or hyperthermia, particularly in very low birth weight (VLBW) infants.

I.3.2 Humidity and oxygen control

Due to immature epidermal barrier, trans epidermal water loss is common among preterm infants. That is why high relative humidity (usually 50–90%) within the chamber of an incubator is used to prevent dehydration and skin breakdown.

Modern incubators have automated systems for humidification that track and adjust humidity continuously. Also, incubators are capable of providing controlled concentrations of oxygen (FiO_2) using integrated gas delivery systems. Oxygen levels are generally kept in a range of 21–40%, depending on the infant’s requirements, and sensor technology help avoid hyperoxia, a recognized risk factor for retinopathy of prematurity (ROP) and chronic lung disease .

I.3.3 Air filtration and airflow mechanisms

In order to reduce nosocomial infections in the neonatal intensive care unit, the incubators are equipped with a HEPA integrated system that filters air with an efficiency of 99.97% for particles that are 0.3 microns or larger. These filters are important for keeping the area surrounding the newborn clean.

Additionally, incubators employ laminar flow systems which guarantee uniform directional movement of air within the chamber. This type of airflow stabilizes temperature and humidity within the system and also reduces microbial contamination by the removal of standby air pockets .

I.3.4 Access ports and ergonomic design

The design of the incubator has been optimized by including numerous flexible access ports, which allow interaction with the infant such as feeding, repositioning, or administering treatment without the need to open the main chamber. This protective feature is effective in preventing unnecessary environmental disturbances like heat loss and infection. Furthermore, the ergonomic aspects of the equipment promote efficient and safe working conditions for healthcare providers. Easy access to patients is made possible through double transparent walls, adjustable working height, and rotating beds. Modern incubators are also equipped with built-in weighing scales, LED lighting, and soundproof materials to reduce the level of stressors experienced by neonates. figure I.8 Neonatal Incubator Components and Functionality provides an overview of these features

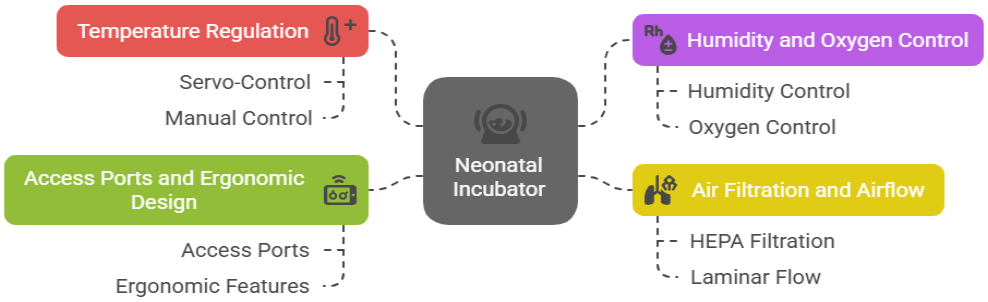


Figure I. 8: Neonatal Incubator Components and Functionality.

I.4 Clinical applications and benefits

I.4.1 Premature infant care

Premature Infant Care. Preterm babies, or babies born before 37 weeks of gestation, have a low maturity that causes physiological problems. These organs affected may include lung, brain, and immunity or the immune system. Specialized clinical care developed to reduce these risks and promote the growth and neurodevelopment [9].

Developmental care programs

Programs such as the newborn individualized developmental care and assessment program (NIDCAP) offer recommended, multisensory interventions that take into account the infant's behavioral cues and expression of stress. These programs lessen the bad effect of the NICU setting by reducing unpleasant stimuli, like too much sound (noise) and bright lights. Clinical trials show that the NIDCAP improves neurobehavioral outcome, enhances autonomic stability and reduces hospital stay of very low birth weight infants (<1250 g).

Kangaroo mother care (KMC)

KMC is continuous skin-to-skin contact of the mother (or father) and the infant. It allows the baby to be warmed, breastfed, and bonded with, also known KMC as Kangaroo Mother Care. KMC decreases mortality by up to 40% in stable low birth weight infants and low incidence of

hypothermia and infection, and is associated with improved growth and neurodevelopment. It also helps confidence of parents and decreases maternal post partum depression. [2]

Nutritional support

To ensure the safe advancement of enteral feeding, it is essential to begin parenteral nutrition early and then advance enteral feeding gradually with breast milk or fortified formula. Breast milk supports gut readiness, immunological protection and bioactive factors which help in reducing NEC [9].

Respiratory support

Ventilatory techniques, for example, CPAP and gentle ventilation strategies establish optimal lung inflation while minimizing lung injury and bronchopulmonary dysplasia [9], The figure I.9 provides an overview.

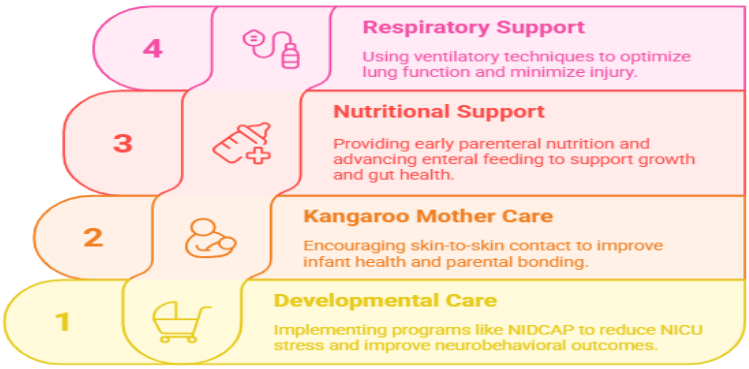


Figure I. 9:Steps to Enhance Premature Infant Care.

I.4.2 Thermoregulation and infection prevention

Preterm babies are at risk of losing heat due to their high surface area to volume ratio, having thin skin and a scarcity of brown fat. This is linked with high morbidity and mortality.

Thermoregulation strategies

- These devices continuously monitor the infant temperature and adjust the ambient heat for maintaining normothermia (36.5–37.5°C).
- Wrapping a baby in plastic film soon after birth without drying helps in preventing loss of heat through evaporating.
- Pre-warmed transport incubators and surfaces to keep ambient temperature.
- Clinical research shows that decreasing admission rates of hypothermia by around 60-70%. Similarly, the need for respiratory support is reduced. Most importantly, mortality rates also lessen.

Infection prevention

Neonatal infections, particularly sepsis and pneumonia, are the leading causes of neonatal deaths in both healthy and at-risk infants, but especially in preterm infants with impaired immune defenses. Key strategies for preventing infections in sick neonates include:

- Strict hand hygiene and the use of barrier precautions: Aimed at limiting nosocomial related infections in NICUs.
- Aseptic techniques during invasive procedures: Ensuing decreased catheter related bloodstream infections.
- Chlorhexidine cord care: It has been shown that chlorhexidine use at the point of cord clamping reduces both bacterial colonization and omphalitis.
- Early and exclusive breast feeding: To provide immunoglobulins and antimicrobial peptides.
- Maternal immunizations: Vaccines for influenza, pertussis and tetanus lower the risk of neonatal infections.
- Antibiotic stewardship: Guidelines for empiric antibiotic therapy based on known pathogens in the local setting and recent resistance patterns should be used and an early, abrupt de-escalation should be used to avoid unwanted resistance. Limited settings are able to rely on simple, validated regimens (e.g. intramuscular gentamicin plus oral amoxicillin) to effectively treat newborn sepsis when hospitalization is not possible [9].

I.4.3 Monitoring and safety features

Timely and accurate monitoring of physiological parameters is necessary for early recognition of deterioration and assisting with decisions for clinical management of premature infants.

Vital signs monitoring

- Heart rate and respiratory rate: For detection of arrhythmias, apnea and respiratory distress via electrocardiography and impedance pneumography.
- Oxygen saturation (SpO2): To help guide the use of oxygen therapy provided via pulse oximetry oxygen therapy avoids issues of hypoxia and hyperoxia which can lead to complications such as: retinopathy of prematurity affecting visual development.
- Temperature (skin or core): To maintain normothermia 24 hours-a-day monitoring available and this is essential in the NICU.
- Blood pressure and Perfusion: To identify hypotension or shock using inclination or non-invasive methods [6].

Alarm systems and safety protocols

NICU monitors have alarms which can be customized based on the patient's status and current guidelines to assist to alert staff when values fall beneath set levels. Devices also can be utilized to integrate the device in an electronic medical record that can flag events as trends and help to build early warning scores. Specific safety protocols in the NICU should include: infection control and prevention policies and protocols, appropriate routine assessment and calibration of medical devices and equipment and implementing an appropriate plan for staff training to be able to minimize errors [9].

Advanced types of monitoring

Emerging technologies such as... Near-infrared spectroscopy (NIRS) of the brain identifies cerebral oxygenation status and transcutaneous monitoring of CO2 allows for identification of the infant's respiratory with reduced and ease of activity on infant [9].

I.5 Challenges and limitations

I.5.1 Noise and environmental stress

Premature infants receiving care in neonatal intensive care units (NICUs) are exposed to a variety of environmental stressors which could, negatively impact their fragile neurodevelopment. The NICU environment is typically one with high levels of noise from alarms, medical equipment, conversations from staff, and aseptic procedures. Noise levels; have been shown to exceed levels recommended as maximums of 45 decibels, report levels similar to those experienced on the street with heavy traffic, or those levels of noise generated from a vacuum cleaner, or between 60 – 70 decibels. The noise could interfere with sleep-wake cycles stimulating increase in heart rates and blood pressure, and in at least some instances elicit stress response to the infantile stressors [7].

In addition to the noise stressors experienced with premature infants are the pain and invasive procedures which can occur on a daily basis, including blood draws, intubation, and intravenous lines. Studies suggest premature infants can undergo between 7.5 and 17 painful and invasive procedures a day while in NICUs, with minimal (if no analgesia/sedation utilized). These infants' nervous system was immature and therefore very sensitive to pain and stress; early experiences can alter long-term pain sensitivity, neuroendocrine function, and architecture of the brain [7].

Compounding environmental stress is bright light, frequent handling, and separate from maternal care. The third trimester of gestation is a critical time in the fetal brain development, including synaptogenesis, and myelination. High levels of sensory stimulation or total sensory deprivation, can disrupt neurodevelopmental trajectories provided that it occurs toward the end of the third trimester. Additionally, prolonged stress can stimulate the hypothalamic-pituitary-adrenal (HPA) axis and raise cortisol levels which can inhibit neuronal development and connectivity [6].

Meeting the above environmental stress with developmental care, included minimizing sound and light, clustering care activities (to sustain sleep), pain management protocol, family-centered care like kangaroo mother care (skin-to-skin contact); it remains difficult to eliminate environmental stressors due to the inherent nature of the NICU and the necessity for ongoing monitoring and interventions [6].

I.5.2 Maintenance and sterilization

In the NICU, infection prevention is important because of the extreme vulnerability of premature babies to nosocomial infections. Premature infants' immature immune systems, defect in their skin barrier, and continuous contact with invasive devices enable opportunistic natural flora and other pathogens to promote infection. The most influential devices initiating a bloodstream infection are central venous catheters and the most common device initiating ventilator-associated pneumonia are endotracheal tubes. Urinary catheters can also contribute to infections as well [6].

Despite the strict sterilization protocols in the NICU, ensuring proper cleaning and disinfection of equipment and surfaces, hand hygiene, and use of barrier precaution/privacy by healthcare workers, can be overwhelming. Staff use equipment such as incubators, ventilators, monitors, intravenous (IV) infusion pumps, and nebulizers, and must constantly be supervising and

assessing the infant, which leads staff to keep high-touch surfaces, medical devices, and themselves clean, and to maintain a sterile environment without interfering with the care of each infant [6].

If equipment is not properly sterilized, healthcare organizations up the risk of outbreaks of multidrug-resistant organisms, which adds to premature infant morbidity, long length of stay, and mortality. Lack of maintenance to cleaning and disinfecting of multidrug-resistant organisms can also lead to improper maintenance of equipment degradation, or improper functioning with a safety risk during use by a patient. Balancing maintaining a sterile environment and providing continuous unperturbed care on a number of infants creates special consideration in the NICU environment, contributing to increased risk of patients. The NICU must have robust training, protocols adapted to resources, and training adequately resourced staff [6].

I.5.3 Resource-limited settings limitations

Resource-limited settings are often considerably limited in the ability to provide comprehensive treatment for patients who are premature. Numerous factors make this extremely difficult to accomplish, including:

- Poor infrastructure: institutions are unable to obtain basic medical supplies (ventilators, servo-controlled incubators, capable continuous cardiorespiratory monitors, etc.) that allow for normothermia and timely resuscitation/evaluation for clinical decline.
- Lack of trained staff: neonatal care is unique and requires training and specialization. under practice is rampant in LMICs not only with neonatologists, but also NEs, nurses, and respiratory therapists.
- Inconsistent access to supplies and medications: Tertiary care often finds sterile catheters low (or not kept on supply), antibiotics are inconsistent (families limited by finances), and parenteral nutrition is inconsistent (either the product is low or management is low). Management/infection control proxies are different than in setting with available supplies.
- Infectious control: overcrowding, sanitation struggles, and no implementation/ability for sterilization or disinfection practices creates a situation for hospital-acquired infections.
- Financial limitations: Families can have shrugs at financial commitments to extend hospital stays and therapies, and then the health system cannot cost-share for seasonal (NICU) time and equipment upgrades.

In order to overcome these shortcomings, recently, simplified, evidence-based interventions have been touted as follows:

- Kangaroo mother care (KMC): An effective, low-cost intervention to improve thermoregulation, breastfeeding, bonding, and ultimately mortality and morbidity [8].
- Simplified antibiotic treatment: For neonatal sepsis, effective intravenous gentamicin and oral amoxicillin (if hospitalization is not possible) was demonstrated [8].
- Simple preventative measures: Emphasizing hand hygiene, clean delivery, chlorhexidine cord care [8].

Nonetheless, a lack of advanced monitoring and respiratory support, in addition, to the inability to support severe, complicated cases, has led to higher mortality and morbidity than in higher-resources environments We also summarize everything we talked about in the (**Figure I.10**).

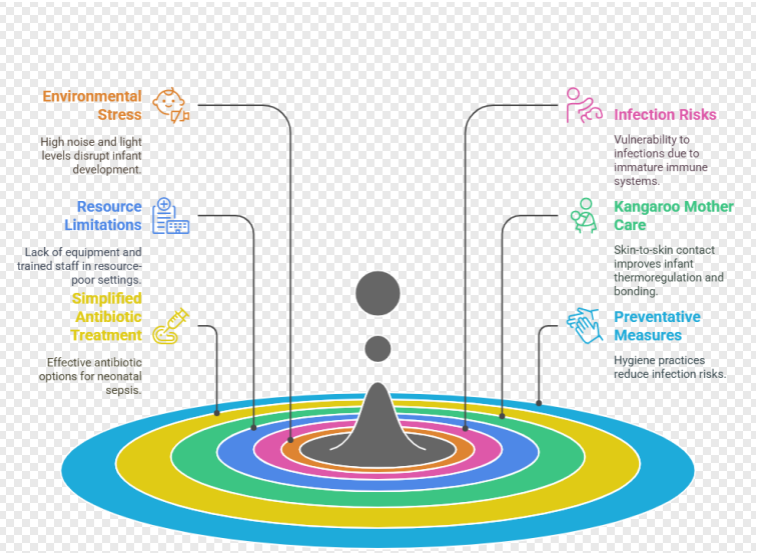


Figure I. 10: NICU Challenges and Solutions.

I.6 Focus on the 8000SC closed incubator model

I.6.1 General presentation of the incubator

The Dräger isolate 8000SC, as shown in **Figure I.11**, is a top-of-the-line neonatal closed incubator that provides a controlled and stable environment for premature and sick newborns. The Isolate features advanced thermoregulation through Dräger's proprietary PID, or proportional-integral-derivative, control algorithm to provide accurate control of air temperatures and skin temperatures. The Isolate also has multiple options for access to the neonate, a height-adjustable table, and developer-friendly touchscreen controls that are suitable for any NICU.

(52-60 in), 104 cm in width (41 in), and 75 cm in depth (30 in), and weight (93 kg/205 lb) without accessories. The unit has a standard hood with front and rear access panels, 6 access ports, and multiple tubing grommets to accommodate clinical interventions while maintaining a controlled environment. The mattress tray size is 40.6 x 81 cm (16 x 32 in) with the ability to tilt the mattress ±12 ° for comfortable positioning [11].

The isolate maintains low airflow velocity (<10 cm/sec) to minimize convective heat loss and to minimize disturbances to the infant. The Isolate uses dual air curtains that maintains a microenvironment while accessing the incubator. There are also optional servocontrolled humidity and oxygen systems that allow the precise regulation of relative humidification (30% to 95%) and oxygen concentration (21% to 65%), which is important for neonatal care.

Safety features include alarms for deviations in temperature, humidity, oxygen concentration, and power outages. The incubator can also trend the data over 24hours and 7 days for ambient and skin temperatures, humidity, O2 levels, heater power, and weight change of the infant, providing data to assist in the clinical decision-making process [11].

The design meets caregivers' needs for comfort and NICU workflow efficiency, including ergonomic foot actuated brakes, smooth running casters for easy mobility, and caregiver maintenance to aid in infection control by enabling sterilization of the humidity system after each patient.



Figure I. 11: DRAGER "8000sc" incubator.

I.6.2 Operating principle

The 8000SC functions under a closed care principle, providing a microenvironment that isolates the infant from temperature changes and airborne infections. The PID control continuously monitors the incubator's air temperature and adjusts it to maintain the setpoint, as close to the desired temperature without exceeding ±0.5 °C of the setpoint. If the incubator has skin temperature probes, it can operate in servo mode: the incubator heats the air based on skin temperature measurements, tracking the infant's skin temperature so overheating or underheating does not happen [11].

Humidity is regulated through a humidification system contained within the incubator which adds water to the air to ensure it can protect the infant's fragile skin and avoid dehydration. The incubator can maintain from 30%82 - 95% relative humidity, and for normal use, the reservoir has a capacity to run high humidity operation for about a day (24 hours).

If oxygen control is installed, the oxygen control will have a servo mode of action capable of maintaining a concentration between 21% to 65% of oxygen, while consistently monitoring the oxygen concentration to avoid hypoxia or hyperoxia due to high or low concentrations of oxygen, which can exacerbate complications like retinopathy of prematurity.

Managing airflow allows for low-velocity circulation (to limit convective heat loss and sound) while dual air curtains at the access ports minimize the intrusion of air from the surrounding environment when the incubator is opened as evidenced by the (figure I.12). The incubator's integrated sensors and alarms provide real-time feedback, as well as safety alarms to users,

allowing caregivers to intervene quickly when the incubator's parameters are outside of safe ranges.

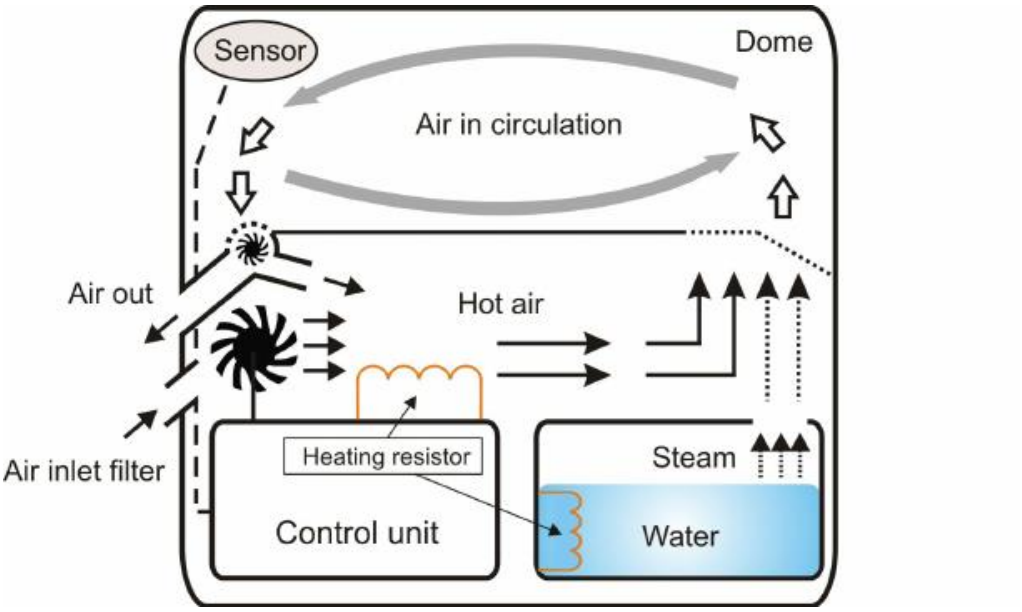


Figure I. 12: General diagram of operation of sensors and actuators.

I.7 Conclusion

In summary, an incubator has gone through a great deal of evolution from an initial, non-adjustable sulfur heater used to warm the air and the infant to a complex incubator that can accurately monitor the environmental parameters and even the patient. As previously noted, the NICU has continued to develop and progress in its understanding and precision of environmental stressors to infants receiving care along with improvement in specialized medical care, thereby improving survival of extremely premature and critically sick infants. Nonetheless, there are challenges the NICU will continue to strive to understand and reduce the environmental impact and overall risks of hospital NICU stay to newborn patient long-term outcomes. The nature of medical care progresses continually and I have no reason to believe that today, there are no unresolved issues with which we grapple or that the newest technology will not progress our clinical categorization; incubators, both closed, open and transport provide a range of care according to different needs of the infant in various clinical contexts.

Chapter II

Overview of Digital Twin Technology

II.1 Introduction

Nowadays, companies are constantly seeking to improve the efficiency of their operations and reduce costs, all while maintaining high levels of quality. To achieve this, they need innovative tools and technologies that allow them to monitor their production and maintenance processes in real time, and to predict potential problems before they occur. This is where digital twins come into play.

In this context, digital twins have become an essential tool for businesses aiming to digitally transform their operations and prepare for the future. By using these technologies, companies can improve efficiency, reduce costs, avoid unexpected downtime, and enhance the quality of their products and services.

In this chapter, we will take a closer look at what digital twins are, how they are used in different sectors, and how they can help companies achieve their performance goals.

From this chapter onward, we will be able to explain what a digital twin is, its types, components, and how it works.

II.2 Definition

A Digital Twin (**DT)** is a dynamic, virtual representation of a physical object or system that accurately mirrors its real-world counterpart throughout its entire lifecycle. It continuously integrates real-time data and leverages simulation, machine learning, and advanced analytics to support decision-making, optimization, and predictive insights

The most common definition of DT was postulated by Glaessgen and Stargel. They dictate that a DT consists of three parts: the physical product, the virtual product, and the communication between them, and define it as follows: “An integrated multi-physics, multiscale, probabilistic simulation of a vehicle or system that uses the best available physical models, sensor updates, fleet history, and so forth, to mirror the life of its flying twin [12].

II.2.1 Overview and functionality of digital twin

We can first summarize the general presentation of the “DT” in figure II.1.

Indeed, the twin, based on data such as drawings (specifications, models, engineering data), productions (method, machine, human resource, machine, etc.), must focus on models that will configure it, such as the various principles (mechanical, engineering), data models (machine learning, deep learning), 3D visualization models, etc. Thus, the combined model and data will allow the twin to provide services that we will call "uses." In this respect, the uses demonstrate a certain degree of maturity.

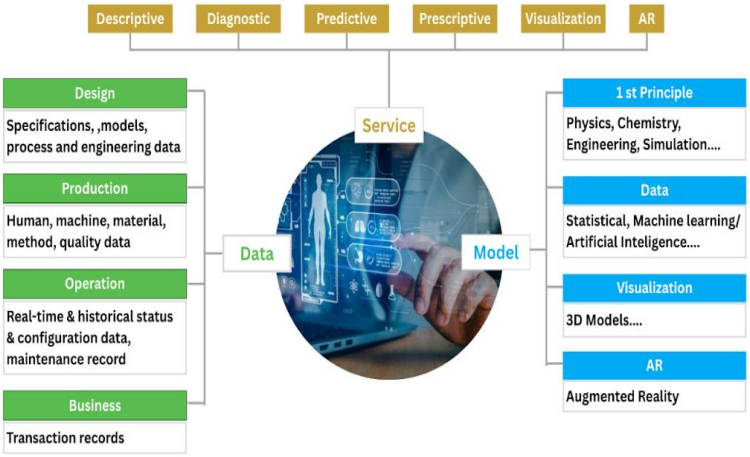


Figure II. 1: Overview of the digital twin.

II.2.2. Digital model

The digital model is defined as a virtual reflection of a physical object, where real-world elements are represented within a digital environment using modeling and simulation techniques. This model is characterized by its comprehensiveness, as it does not only represent the mechanical structure or geometric dimensions, but also includes all technical and functional aspects related to the system or product. This includes, for example, electronic systems, electrical connections, source code, firmware, as well as information related to production, operation, and maintenance.

In this early phase, the digital model serves as a fundamental tool in design and analysis, primarily used to visualize the product or system, test its performance, or simulate its operating conditions. Common examples of these models include 3D representations used in fields such as mechanical engineering or industrial design, which provide an accurate visualization of objects even before they are physically manufactured.

This type of model provides a rich informational base that can be utilized in design, development, and testing phases. It is also used in performance analysis and predicting potential issues before the system is implemented in the real world. Thanks to its comprehensive nature, the digital model becomes an effective tool for decision-making and improving the quality of industrial or technical processes. However, this model remains a static representation, disconnected from the physical object, meaning that it does not interact with or change based on real-time physical changes. As such, its data updates rely entirely on manual human

intervention, rather than on live data flows or direct real-time sensing from the physical environment.

II.2.3 Digital twin vs. digital model

Table II. 1 the difference between digital twins and digital model

Aspect	**Digital Model**	**Digital Twin**
Definition	A static digital representation of a physical object or system, simulating its basic characteristics	A dynamic, continuously updated digital replica of a physical object or system that interacts in real time
Interactivity	No interaction with the physical world; updates are done manually	Fully interactive with the physical counterpart through two-way communication
Data Source	Manually entered by engineers or users	Real-time data acquired through sensors and IoT systems
Connection to Physical World	Not connected	Permanently and actively connected via smart networks
Real-Time Capability	Not supported	Fully supported
Analytics & Prediction	Basic analytics only	Supports predictive analytics and autonomous decision-making via AI
Self-Learning Capability	Not available	Enabled through AI and machine learning integration
Accuracy and Adaptability	Limited accuracy; does not reflect real-time changes	High accuracy; dynamically adapts to changes in the physical environment
Main Uses	Design, simulation, visualization	Smart monitoring, predictive maintenance, performance optimization, virtual testing
Lifecycle Coverage	Mainly used in early product development stages	Covers the entire lifecycle—from design to operation, maintenance, and end-of-life
Flexibility and Responsiveness	Inflexible; requires manual updates	Highly responsive; automatically adapts based on sensor input

Cost and Implementation Effort	Lower cost; relatively easy to implement	Higher cost; requires advanced digital infrastructure and analytics
Examples	A 3D CAD model of an engine without operational data	A digital twin of a neonatal incubator receiving real-time temperature, humidity, and pressure data

II.3 Development and history of digital twin

The origin of the DT is attributed to Michael Grieves and his work with John Vickers of NASA, with Grieves presenting the concept in a lecture on product life-cycle management in 2003. In a time when Grieves describes virtual product representations as "relatively new and immature" and data collected about physical products as "limited, manually collected, and mostly paper-based", Grieves and Vickers saw a world where a virtual model of a product would provide the foundations for product life-cycle management. The initial description defines a DT as a virtual representation of a physical product containing information about said product, with its origins in the field of product life-cycle management. In an early paper [13].

NASA's Apollo space program was the first program to use the 'twin' concept. The program built two identical space vehicles so that the space vehicle on Earth can mirror, simulate, and predict the conditions of the other one in space. The vehicle that remained on Earth was the twin of the vehicle that executed a mission in space [14].

Even though DT technology has gained massive popularity in the past couple of years, the concept is not entirely new. Its concept came into being with PLM in 2002 at the University of Michigan by Michael Grieves [15]. The proposed model has three components: real space, virtual space, and a linking mechanism for the flow of data/information between the two; the model was then referred to as the mirrored Spaces Model [16].

The concept of the DT has undergone a fascinating evolution, progressing from its initial inception to becoming a transformative force in modern technology and industry. This timeline highlights key milestones and influential contributors that have shaped the concept's definition and application.

- In 2003, Professor Michael Grieves proposed the concept of virtual digital representations equivalent to physical products in a product PLM course at the University of Michigan and gave it a definition: which is a digital copy of one or a set

of specific devices that can abstractly represent a real device and can be used as a basis for testing under real or simulated conditions [13].

- In 2006, the US National Science Foundation (NSF) first proposed the concept of the information physics system CPS, which could also be translated into a network entity system, or an information physical integration system [17].
- In 2011, Professor Michael Grieves, in his book "Virtually Perfect: Driving Innovative and Lean Products through PLM" [18].
- After 2011, the DT ushered in a new development opportunity. The DT was proposed and further developed by the US Air Force Research Laboratory in 2011; the aim is to solve the maintenance and life prediction of aircraft in the future complex service environment [19].
- In 2012, in the face of future aircraft light quality, high load, and the demand for longer service time under more extreme environments, **NASA** and the **US** Air Force Research Laboratory in cooperation put forward the common **DT** example of future aircraft. For aircraft, flight systems, or launch vehicles, they define DT as an integrated multi-physical, multi-scale, probabilistic simulation model for aircraft or system, which uses the best available physical models, updated sensor data, and historical data to reflect the state of the flying entity corresponding to the mode [20].
- In 2013, Germany proposed "Industry 4.0," whose core technology is CPS. CPS is a multidimensional complex system that is integrated with computing, communication, control, network, and the physical environment[21].
- In 2014, Professor Michael Grieves elaborated on DT in detail in his white paper "DT: Manufacturing Excellence through Virtual Factory Replication" [22].
- In 2015, General Electric Company planned to implement real-time monitoring, timely inspection, and predictive maintenance of engines based on digital hygiene and through its cloud service platformz- Predix, using advanced technologies such as big data and the IoT [23].
- In 2017, to realize the interactive fusion of the physical world and the information world of the manufacturing workshop, Tao Fei put forward the realization mode of the DT workshop and made clear its system composition, operation mechanism, characteristics, and key technology, which provided the theory and method reference for the realization of the information physical system of the manufacturing workshop [24]. figure II.2 depicts the evolutionary path of the DT concept over time.

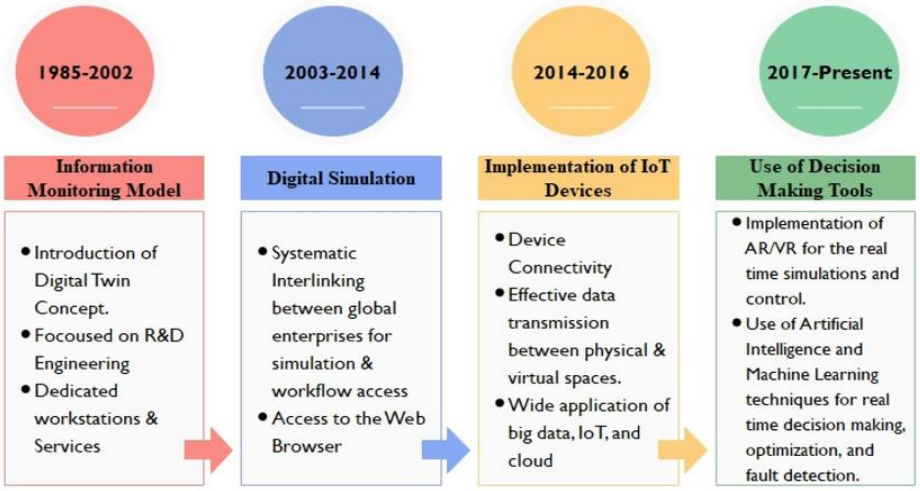


Figure II. 2: Evolution of digital twin.

II.4 Types of digital twin

The different types of digital twins can be grouped into four main categories [25].

II.4.1 Asset twin

Asset twins describe how individual elements work together as a whole. On the other hand , they allow you to validate the entire system. They can be used to verify compatibility between individual components.

II.4.2 System twins or unit twin

Operate at a higher level. They combine various individual asset twins and offer you the ability to examine how these individual twins work together.

II.4.3 Process twin

Can be used to simulate the entire operation. Only at this level does the full complexity of monitoring via digital twins become evident. After all, a process only becomes functional and efficient when all units, assets, and components fulfill their purpose.

II.4.4 Component twin

Component twins represent each individual element of a system, whether it be a hardware component, a piece of equipment, or a machine. They are a perfect survivor and simulator of a similar component, as well as the interactions within the global system.

II.5 Components of a digital twin

Figure II.3 highlights the rotation of the data flow between, on the one hand, the physical part,which consists of sensors and actuators, and on the other hand, the digital or virtual part,which is constantly updated [26]. The above elements are explained as follows:

II.5.1 Sensors

Distributed throughout the manufacturing process, they create signals that allow the twin to capture operational and environmental data related to the physical process in the real world.

II.5.2 Data

Real-world operational and environmental data are aggregated from sensors and combined with organizational data.

II.5.3 Integration

Sensors provide data to the digital world through integration technology between the physical and digital worlds.

II.5.4 Analytics

Analytical techniques are used to analyze data through computer simulations and visualization processes used by the digital twin to generate insights.

II.5.5 Actuators

If an action is warranted in the real world, the digital twin produces the action using actuators, subject to human intervention, which triggers the physical process.

II.6 Optimal applications and use cases for digital twins

DT offers valuable advantages in the maritime and shipping industries, particularly when it comes to design customization. By creating virtual replicas of vessels, designers and engineers can simulate and test different design configurations, materials, and systems. This allows for thorough analysis and optimization of ship designs before physical construction. DT also provides insights into the ship's performance under various operating conditions, aiding in the identification of potential issues and the implementation of necessary improvements. Ultimately, DT enhances the design process, leading to more efficient and tailored maritime solutions.

Digitalization has become a vital part of the maritime industry, ultimately aimed at making the industry more innovative and efficient [27]. An example of the use of DT in Maritime and Shipping is shown in figure II.3.

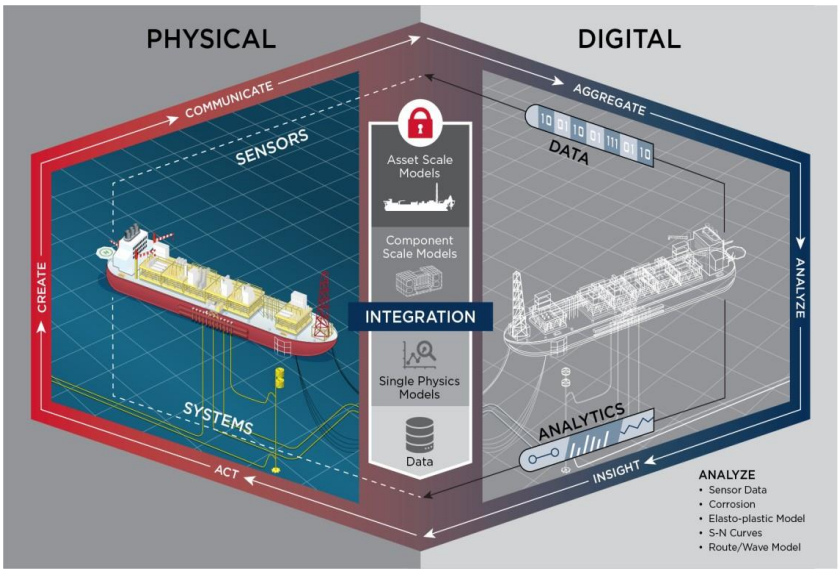


Figure II. 3: Digital Twin in Maritime and Shipping.

II.6.1 Manufacturing

DT plays a crucial role in manufacturing, specifically in the areas of product development and predictive manufacturing. By creating virtual replicas of products, manufacturers can thoroughly test and optimize designs before physical production. This allows for the identification of potential issues early on, reducing time-to-market and improving product quality. Additionally, DT enables predictive manufacturing, where real-time data from sensors and production processes is fed into the virtual model to monitor and predict performance, identify bottlenecks, and optimize production efficiency. This helps manufacturers make data-driven decisions, improve productivity, and reduce downtime. Ultimately, DT enhances both the product development process and manufacturing operations in the industry.

In manufacturing, the DT serves as a virtual replica of a physical product, incorporating real-time process data and analytics to accurately reflect the product's design, construction, and maintenance. This operational context is crucial for optimizing performance. While virtual models are conceptual, real-time operational data provides a digital representation of actual physical events. Computer-aided design models capture the virtual characteristics of the DT's physical counterpart. However, to enable analytics applications and enhance the performance-based DT, real-time operational and asset data are necessary to define its state and behavior, facilitating optimization and process improvement [28]. An example of the use of DT in the manufacturing field is shown in figure II.4.

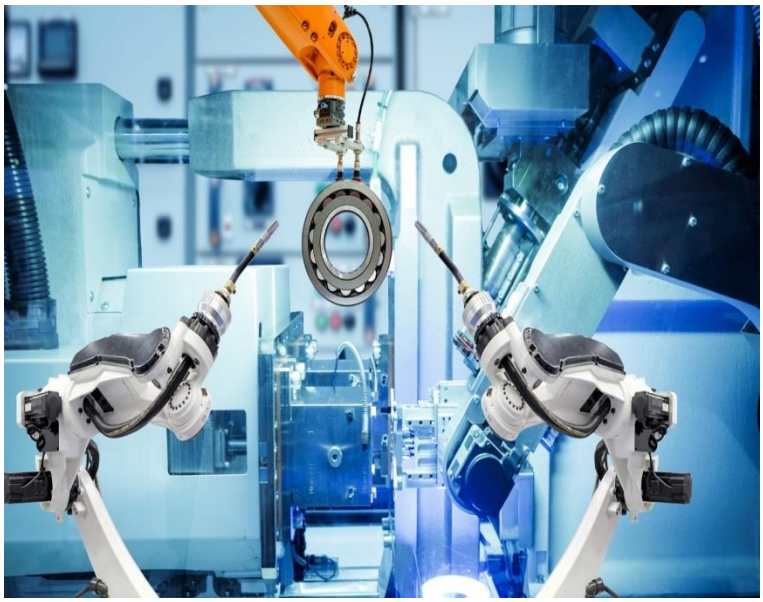


Figure II. 4: digital twin in manufacturing.

II.6.2 City Management

DT is a powerful tool for modeling and simulating smart cities. It allows city managers to create virtual replicas, incorporate real-time data, and run simulations to optimize urban systems, enhance decision-making, and foster collaboration among stakeholders. By leveraging the capabilities of DT, city management can effectively plan, monitor, and improve the overall efficiency, resilience, and sustainability of smart cities. The emergence of Industry 4.0, incorporating advanced sensors, AI, and IoT technology, has significantly expanded the applications of DT now in city and infrastructure management. DT now offers virtual representations of equipment, processes, workflow information, and digital data derived from various sensors, all synchronized in real-time. Moreover, the integration of DT with AI and big data enables the optimization of infrastructure operations, leading to cost and time reductions in operations and maintenance by implementing predictive maintenance strategies [29].

An example of the use of DT in City Management is shown in figure II.5.

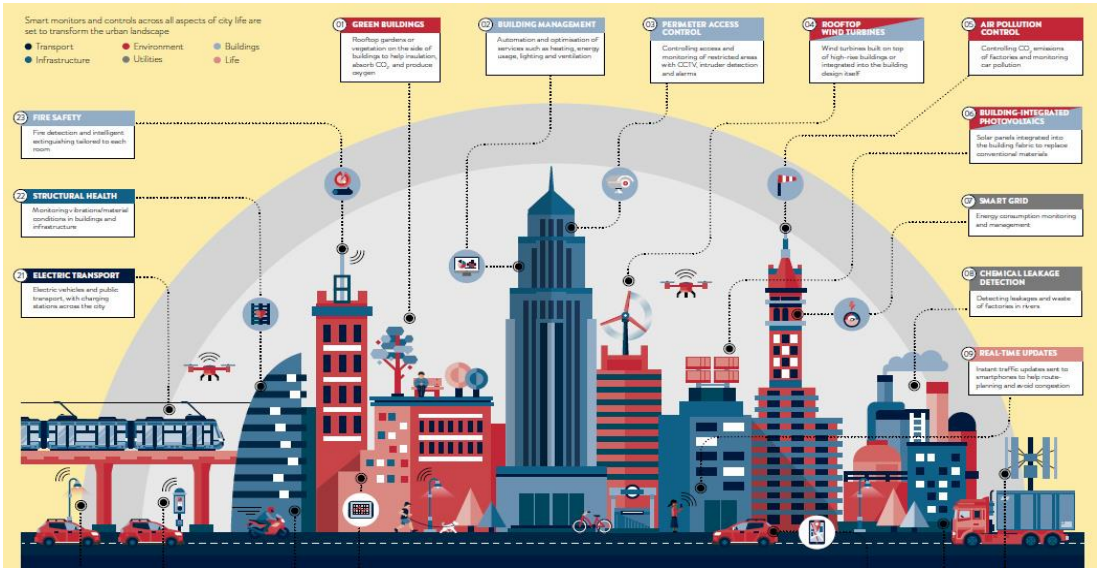


Figure II. 5: Digital twin city management.

II.6.3 Aerospace

In the aerospace industry, the concept of a DT is increasingly utilized to enhance the maintenance and performance of aircraft. By leveraging advanced technologies and predictive analytics, DT technology allows aerospace professionals to create virtual replicas of physical aircraft, enabling them to anticipate and address potential problems before they occur. An illustration demonstrating the utilization of DT in aerospace is depicted in figure II.6.



Figure II. 6: digital twin in aerospace.

DT plays a significant role in the aviation industry by offering virtual, three-dimensional models of real aircraft or specific parts such as engines. Prominent aviation manufacturers, including Boeing, have integrated DT into their avionics processes. These DTs enable engineers to conduct diverse experiments using virtual representations of airplanes, allowing them to anticipate both positive and negative outcomes. By leveraging IoT location and sensor data,

engineers can expose the DT aircraft to simulated weather conditions and environments that are expected to be encountered in real-world scenarios [30].

II.6.4 Healthcare

DT offers significant benefits to the healthcare industry, particularly in terms of improving operational efficiency. By creating virtual replicas of healthcare facilities, administrators and managers can simulate and analyze different scenarios to identify areas for improvement. This includes optimizing patient flow, resource allocation, and staff scheduling. DT also enables process optimization, such as simulating surgical procedures to enhance precision and reduce errors. Overall, DT has the potential to streamline healthcare operations and improve patient care.

Believe that the use of DT technology will become more prevalent in the medical field to address various issues that traditional methods cannot fully explain. It enables real-time monitoring, dynamic analysis, and precise treatment for diseases. Modeling the interactions between the DT and the physical entity in a medical environment has the potential to revolutionize healthcare. While there are technical and ethical challenges to overcome, progress in DT technology is promising. Its application in medicine extends beyond diagnosis and treatment to include health prediction, providing a qualitative understanding of health and disease. DT healthcare represents the future of medicine, enabling precise and personalized treatments to become a reality [31]. An example of the use of DT in the medical field is shown in figure II.7.

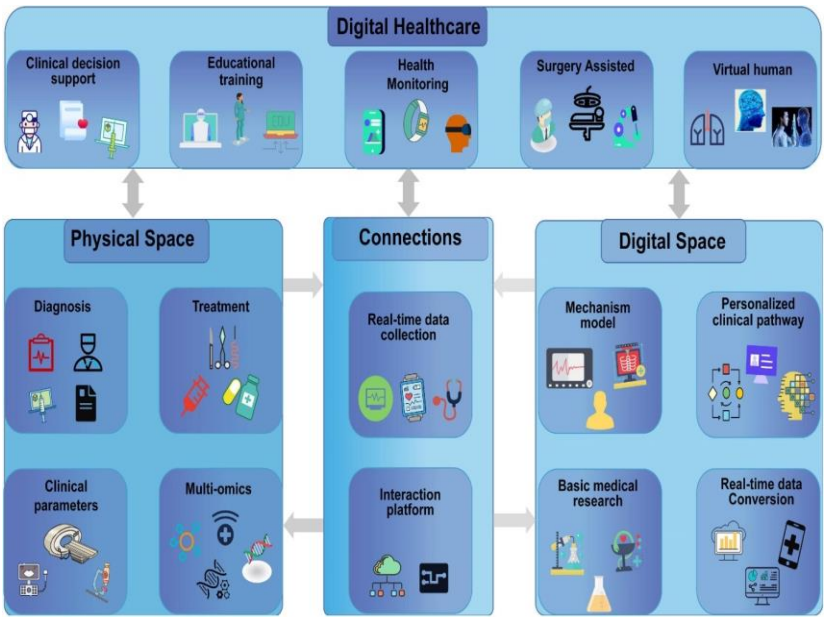


Figure II. 7:Application and prospects of digital twin technology in the medical field.

II.6.5 Agriculture

Digital twin technology is gaining increasing interest in the agricultural sector, as it enables the creation of accurate digital models that represent physical systems such as crops, land plots, and agricultural equipment figure II.8. These models rely on data collected from sensors and various sources to simulate and analyze the behavior of these systems, helping farmers make more informed decisions.

The applications of digital twins in agriculture are diverse, including crop modeling, resource management (such as water and fertilizers), monitoring of diseases and pests, and planning agricultural operations. By simulating various scenarios, farmers can optimize their practices to increase yield, reduce losses, and enhance both environmental and economic sustainability.



Figure II. 8: The Connected Farmer: A Digital Twin Approach to Smart Agriculture.

II.7 Key benefits of digital twins

The use of digital twins offers numerous logistical advantages that improve the profitability of the entire supply chain.

- **Enhanced decision-making:** Digital twins facilitate informed decision-making by providing real-time data and predictive insights. This capability allows organizations to anticipate issues and implement solutions proactively.
- **Improved product quality and innovation:** By simulating and testing products in a virtual environment, digital twins enable continuous improvement and innovation, leading to higher quality outcomes.
- **Reduced time to market:** The ability to model and test products virtually accelerates the development process, reducing the time required to bring products to market.

- **Operational efficiency and cost savings:**Digital twins optimize operations by identifying inefficiencies and predicting maintenance needs, resulting in cost reductions and improved resource utilization.
- **Enhanced customer satisfaction**: By ensuring product reliability and enabling personalized services, digital twins contribute to higher customer satisfaction levels.

II.8 Challenges

While implementing predictive maintenance through a digital twin offers many benefits, its deployment can pose the following challenges:

A digital twin must accurately reflect all the properties of a physical twin, including mechanical (suction pressure, design temperature, etc.) and electrical (capacitance, conductivity, etc.) properties.

Detailed plans of machine failures are required. To predict failures, a digital twin must be fed data on equipment failure modes. This data must be collected over an extended period of time to observe a machine throughout its degradation process.

A digital twin requires redesign with any change in the configuration or condition of the equipment element. Any change affecting equipment performance requires a change in its model and underlying algorithms. Such changes at the machine or factory level are not always reflected in factory specifications and therefore cannot be simulated accurately, increasing the risk of errors.

Although deploying predictive maintenance based on digital twins is time-consuming and labor-intensive, the technology offers the ability to timely identify disruptions in asset performance, predict potential problems, and simulate various maintenance scenarios. It helps companies eliminate machine downtime, reduce equipment maintenance costs, improve equipment reliability, and extend its lifespan. The complexity of using artificial intelligence in industrial automation requires manufacturers to collaborate with specialists to find customized solutions. To achieve this, several elements must be considered to meet the implementation needs:

- Historical data collection
- Live data capture via sensors
- Data aggregation
- Connectivity via communication protocols Integration with automation
- AI applications: machine learning and other techniques

II.9 Digital twin: an integration of interconnected technologies

The digital twin is built upon the convergence of multiple interdependent technologies, forming a complex and intelligent ecosystem:

- **Mathematical and Visual Modeling (2D/3D):** These models enable the accurate representation of the physical entity's structure and behavior within a virtual environment, serving as a foundation for simulation and analysis.
- **Industrial Internet of Things (IIoT):** By integrating IoT with cloud infrastructure, IIoT facilitates machine-to-machine (M2M) communication and ensures seamless bidirectional data exchange between the physical world and its digital counterpart.
- **Artificial Intelligence (AI):** Through machine learning and deep learning algorithms, AI enhances the predictive and decision-making capabilities of digital twins, progressing toward cognitive, self-adaptive systems.
- **Big Data, Smart Data, and Lean Data:** These approaches support the collection, processing, and intelligent exploitation of large-scale data generated by physical systems, enabling deep and context-aware analysis.
- **Enterprise Software (CRM, ERP, MES, etc.):** These systems allow for manual or semi-automated control and monitoring. When integrated with IoT, they strengthen human-machine interaction (HMI), increasing operational agility and responsiveness.

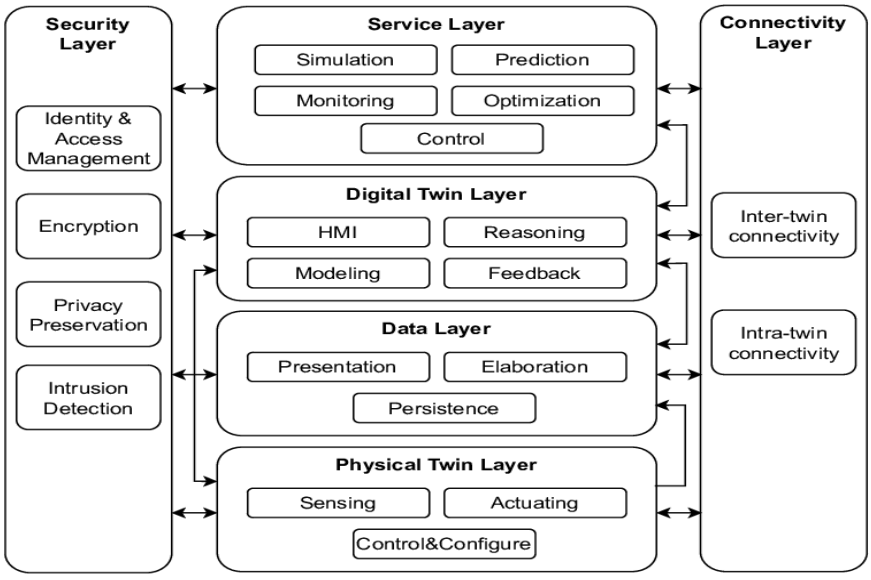


Figure II. 9: Layers of the Digital Twin.

II.10 Different layers of the digital twin

Digital twins are virtual representations of physical objects and systems in real time. They consist of three layers (Figure II.9) that interact to provide a complete and accurate version of the object or system. Here are the main layers of digital twins:

II.10.1 Physical layer

This layer represents the real, tangible aspect of the object, system, or process being mirrored by the digital twin. It includes hardware components such as equipment, machines, physical infrastructure, sensors, and monitoring devices. Data from this layer is collected through embedded sensors and other measurement tools operating in the field.

II.10.2 Digital layer

This is the virtual representation of the physical layer. It uses digital models and simulations to replicate the behavior, characteristics, and interactions of the real-world entity. This layer contains data such as geometric models, physical properties, operational algorithms, and associated software.

II.10.3 Communication Layer

This layer plays a critical role in enabling two-way communication between the physical and digital layers. It facilitates the collection of real-time data from the physical world and allows transmission of insights, commands, or updates from the digital model back to the physical system. It relies on technologies like the Internet of Things (IoT), wireless networks, and various communication protocols.

II.11 Digital twin platforms

Digital Twin Platforms are comprehensive software environments that enable the creation, management, and operation of digital twins. These platforms serve as a bridge between the physical and digital worlds by integrating real-time data from IoT devices, sensors, and enterprise systems. Their main goal is to provide a dynamic virtual representation of physical assets, systems, or environments, allowing users to monitor performance, predict failures, and optimize operations.

Several platforms have emerged as leaders in this domain, each offering unique features and targeting different industries:

II.11.1 AWS IoT twin maker

AWS IoT Twin maker is an AWS IoT service that you can use to build operational digital twins of physical and digital systems. AWS IoT Twin Maker creates digital visualizations using measurements and analysis from a variety of real-world sensors, cameras, and enterprise applications to help you keep track of your physical factory, building, or industrial plant. You can use this real-world data to monitor operations, diagnose and correct errors, and optimize operations [32].

II.11.2 Azure digital twin

Azure Digital Twins is a platform as a service (PaaS) offering that enables the creation of twin graphs based on digital models of entire environments, which could be buildings, factories, farms, energy networks, railways, stadiums, and more

Azure Digital Twins can be used to design a digital twin architecture -even entire cities. These digital models can be used to gain insights that drive better products, optimized operations, reduced costs, and breakthrough customer experiences. that represents actual IoT devices in a wider cloud solution, and which connects to IoT Hub device twins to send and receive live data[33].

II.11.3 Eclipse ditto:

Eclipse ditto™ is a technology in the IoT implementing a software pattern called “DT”. A digital twin is a virtual, cloud based, representation of his real world counterpart (real world “Things”, e.g. devices like sensors, smart heating, connected cars, smart grids, EV charging stations, …).

The technology mirrors potentially millions and billions of digital twins residing in the digital world with physical “**Things**”. This simplifies developing IoT solutions for software developers as they do not need to know how or where exactly the physical “Things” are connected.

With Ditto a thing can just be used as any other web service via its digital twin[34].

II.11.4 Siemens mindsphere

Siemens mindsphere is an open, cloud-based IoT operating system that enables companies to connect, monitor, and analyze their assets and processes in real-time. It provides a platform for data-driven insights and optimization, enhancing efficiency, sustainability, and security across various industries. For detailed information, refer to the search results provided[35].

II.11.5 PTC thing worx

ThingWorx is an IoT platform developed by PTC that enables the connection of devices to cloud-based applications for monitoring and data analysis. The platform supports the creation of digital twins for industrial machines and equipment, as well as for production and manufacturing environments. These platforms are essential for implementing digital twin technology in smart

II.12 Conclusion

n conclusion, this chapter has provided a comprehensive overview of Digital Twin (DT) technology and its evolution, while highlighting its increasing relevance in healthcare environments particularly in the management of neonatal incubators. Digital Twins, as virtual representations of physical systems, offer critical functionalities such as real-time monitoring, predictive analysis, and intelligent decision support. These capabilities are essential for optimizing neonatal care, where precision, responsiveness, and safety are vital.

Driven by advancements in cloud computing, big data analytics, and the Internet of Things (IoT), Digital Twins have evolved into powerful platforms capable of simulating and managing complex medical environments. In the context of neonatal incubators, DT technology enables continuous monitoring of vital signs, environmental conditions, and device performance, thereby enhancing clinical outcomes and reducing risks.

Ultimately, leveraging Digital Twin platforms in neonatal incubator management represents a transformative step toward intelligent, patient-centered healthcare. It empowers medical professionals with deeper insights, improves system resilience, and contributes to the broader goals of Industry 4.0 in the healthcare domain.

Chapter III

Software and Hardware Tools

III.1 Introduction

In developing a reliable Digital Twin (DT) for neonatal incubator management, selecting the right software and hardware tools is essential. This chapter outlines the technical ecosystem supporting the DT system, which integrates real-time monitoring, prediction, and visualization. The software stack includes Python (in VS Code), Pandas, NumPy, a React web app, Blender for 3D modeling, an LSTM network for predictive analytics, and Firebase Cloud for data storage and synchronization. On the hardware side, the system relies on an ESP32 microcontroller connected to environmental sensors, enabling real-time data collection and interaction with the virtual model. These components together ensure seamless synchronization between the physical incubator and its digital counterpart.

III.2 Software tools

III.2.1 Programming language – python

Python is a powerful and easy-to-learn programming language. It features high-level data structures and allows for a simple yet effective approach to object-oriented programming. Because of its elegant syntax, dynamic typing, and interpreted nature, Python is an ideal language for scripting and rapid application development in many domains and on most platforms [36].

The main libraries used include**:**

- **NumPy and pandas:** For numerical and time-series data processing.
- **Matplotlib and seaborn:** For data visualization and plotting graphs.
- **Scikit-learn:** For data splitting, model evaluation (e.g., RMSE, R^2), and auxiliary machine learning tasks.
- **Tensorflow / Pytorch:** For developing the LSTM model used in temperature and humidity prediction.

III.2.2 Visual studio code (VS Code)

Visual Studio Code (VS Code) is a powerful, cross-platform code editor developed by Microsoft, widely used for developing web and cloud-based applications. It combines performance and flexibility, offering developers a smooth and modern interface to write, manage, and debug code efficiently

It supports numerous programming languages, including **C++, Python, Java, HTML, JavaScript,** and many others, making it suitable for a wide range of development tasks. Additionally, it provides support for **Node.js, ASP.NET**, and other popular frameworks.

VS Code allows users to manage their projects effectively by integrating essential tools such as **Git version control** and a built-in terminal. It also facilitates debugging, editing, and organizing code with features like split-screen views and syntax-aware editing. The editor is highly **customizable**, supporting extensions and themes to suit different workflows and development styles.

III.2.2.1 Key features of visual studio code

Here are the key features of VS Code

Cross-Platform Support: Available on Windows, macOS, and Linux, allowing developers to work seamlessly across different operating systems.

1. **Lightweight and Fast**: Designed to be quick and responsive without consuming too many system resources.
2. **Built-in Git Integration**: Provides native support for Git version control, enabling easy management of repositories within the editor.
3. **Extensive Extension Marketplace**: Offers thousands of extensions to add language support, debuggers, themes, and other tools to customize and enhance the editor.
4. **IntelliSense**:Features intelligent code completion, syntax highlighting, and real-time error detection to improve coding efficiency.
5. **Integrated Debugging:** Includes a powerful debugger for multiple languages to test and troubleshoot code directly inside the editor.
6. **Customizable Interface**: Allows users to personalize the editor's look, keyboard shortcuts, and settings to fit their workflow.
7. **Multi-language Support**: Supports a wide range of programming languages out-of-the-box and via extensions, it can accommodate many others, including Python, C++, Java, and Go.
8. **Terminal Integration**: Embedded terminal to run command-line tasks without leaving the editor.
9. **Source Control Management**: Simplifies working with source control systems, especially Git, directly from the editor interface

III.2.2.2 teps to create a new project inVS Code

A. Package installation instructions

1. The latest installation package should be downloaded from the official download page corresponding to the operating system in use (figure III.1).
2. The downloaded installation file should then be extracted.
3. After extraction, the directories and files should be copied to the appropriate folder.
4. The extracted installation file should be executed to complete the installation.

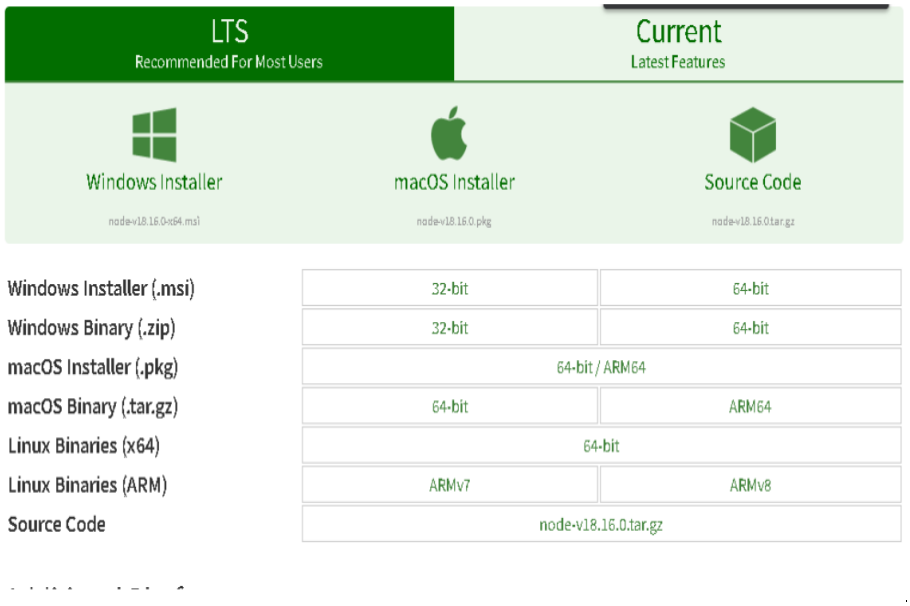


Figure III. 1: VS code Official download page.

B. Project folder setup

A folder intended for the project should be created and opened using VS Code. This can be done by selecting **'Open Folder'** from the interface.

- VS Code should first be launched.
- From the top menu, the following navigation should be used: **File → Open Folder**
- The folder previously created (e.g., *MyProject*) should be selected and opened.
- Once opened, VS Code will recognize this folder as a project workspace.

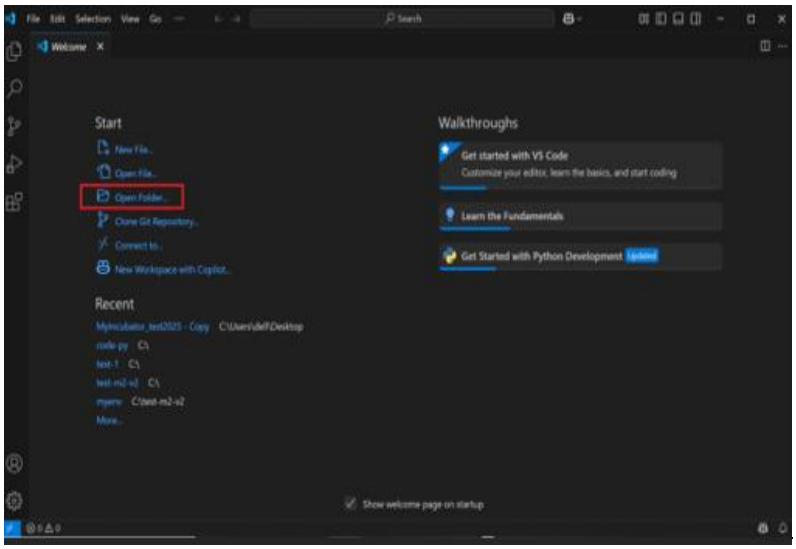


Figure III. 2: Software VS Code.

C. React project initialization

- To initialize a new React project, the following command should be executed in the terminal:

  npx create-react-app client
- This will create a new project named *client* using Create React App.
- Once the creation process is complete, navigation to the project directory should be done using the command:

  cd client
- To install the required dependencies, the following command should then be executed:

  npm install three @react-three/fiber @react-three/drei
- This command ensures that the libraries *three*, *@react-three/fiber*, and *@react-three/drei* are downloaded from the npm registry and added to the package.json file as project dependencies.
- Upon successful installation, two new component files and a configuration file named firebase.js should be created within the project structure.

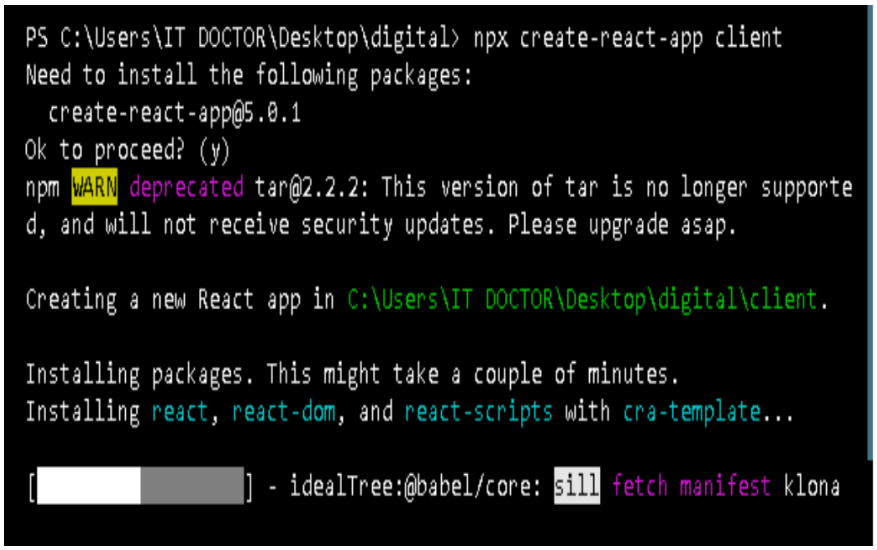


Figure III. 3: Creating a REACT project.

D. Project organization

For better project management, appropriate subfolders such as src/, data/, models/, etc., should be created to organize the source code and assets.

A VS code folder will be automatically generated by VS Code to store workspace-specific settings.

III.2.3 Blender

Blender is the free and open source 3D creation suite. It supports the entirety of the 3D pipeline-modeling, rigging, animation, simulation, rendering, compositing and motion tracking, even video editing and game asset creation. Advanced users employ Blender's API for Python scripting to customize the application and write specialized tools; often these are included in Blender's future releases. Blender is well suited to individuals and small studios who benefit from its unified pipeline and responsive development process. Examples from many Blender-based projects are available in the showcase.

Blender is cross-platform and runs equally well on Linux, Windows, and Macintosh computers. Its interface uses OpenGL to provide a consistent experience. To confirm specific compatibility, the list of supported platforms indicates those regularly tested by the development team [37].

A. User Interfac

Blender provides a modular and flexible interface composed of panels, menus, and toolbars. The default interface layout is shown in figure III.4**.**

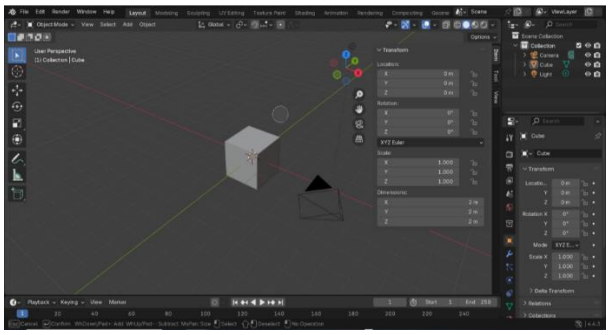


Figure III. 4: Initial Blender Interface.

B. 3D Modeling

Blender supports modeling with basic geometric shapes (e.g., cubes, spheres, cylinders) as well as advanced features such as sculpting, smoothing, and curve-based modeling see figure III.5.

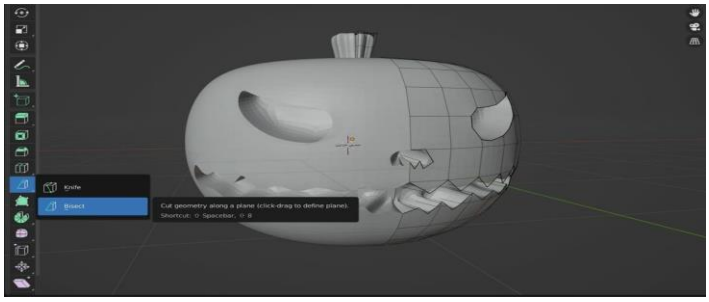


Figure III. 5: Modeling Tools in Blender.

C. Materials and Texturing

Materials can be defined using a node-based system that allows the combination of shaders, textures, and visual effects. This enables the creation of photorealistic or stylized surfaces **(**figure III.6**).**

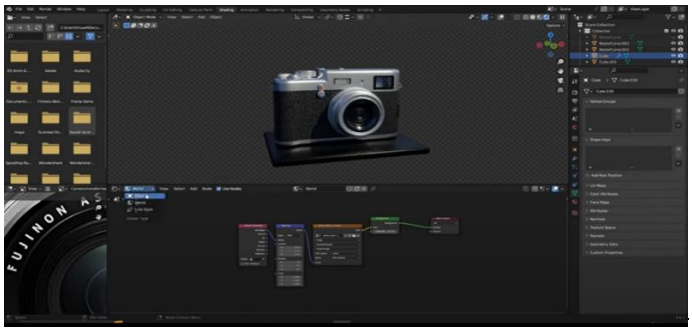


Figure III. 6: Material Creation in Blender.

D. Animation and Simulation

Blender includes a robust animation framework for keyframing, motion paths, and object interaction. It also supports simulations such as rigid bodies, fluids, and particle systems. The animation process is shown in figure III.7**.**

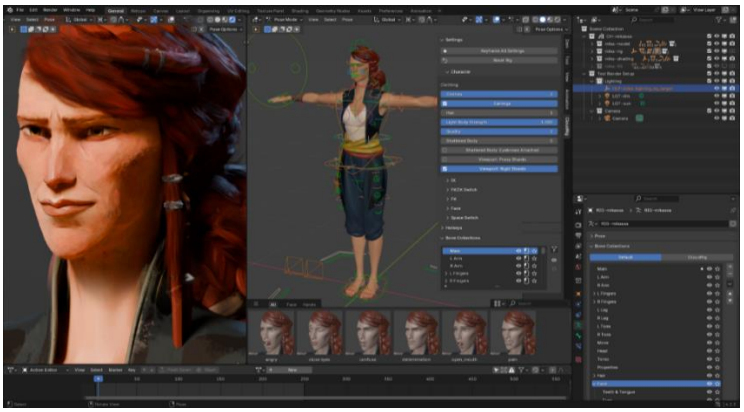


Figure III. 7: Blender Animation Workflow.

E. Lighting and Rendering

Lighting configurations include point lights, spotlights, and directional lights. Once configured, scenes are rendered using Blender's internal or external engines to produce final images or animations **(**figure III.8**).**

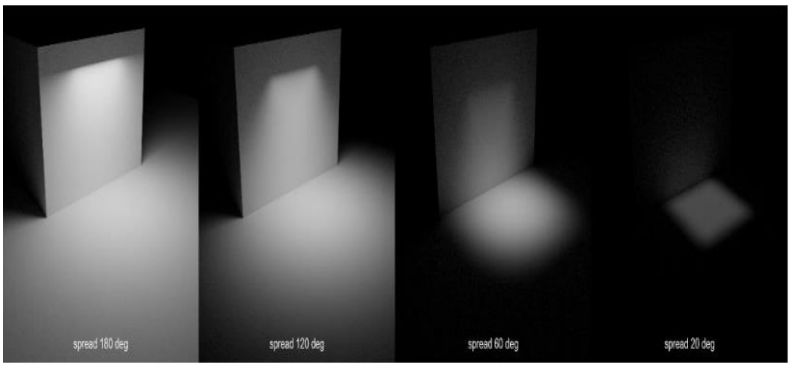


Figure III. 8: Lighting Tools in Blender.

F. Video Editing and Compositing

Blender includes a video sequence editor for cutting, trimming, and compositing video clips. Features include effects layers, transitions, text overlays, and color adjustments.

G. Scripting and Customization

The Python scripting API allows users to automate tasks and build custom features. A sample script interface is shown in figure III.**9.**

```python
import bpy
bpy.ops.object.select_all(action='DESELECT')
bpy.context.view_layer.objects.active = None

for obs in bpy.context.scene.objects:
    if obs.name in bpy.context.view_layer.objects:
        if obs.type == 'MESH':
            bpy.ops.object.select_all(action='DESELECT')
            nocolon = obs.name.rsplit(maxsplit=1, sep=':')
            print ('nocolon pass done for ' + nocolon[0])
            tname = nocolon[0].rsplit(maxsplit=1, sep='.')
            print ('nodot pass done for ' + tname[0])
            bpy.ops.object.select_pattern(pattern= tname[0]+'*', case_sensitive=False, extend=True)
            print ('selection pass done for ' + tname[0])
            obs.select_set(True)
            print ('selection set done for ' + tname[0])
            bpy.context.view_layer.objects.active = obs
            print ('set active object done for ' + tname[0])
            bpy.ops.object.join()
            print ('join pass done for ' + tname[0])
            print ('iteration done for ' + tname[0])
            bpy.context.view_layer.objects.active = None

        else:
            print(obs.name + ' is not a MESH.')

    else:
        print('There is no ' + obs.name + ' in view_layer')
        bpy.data.objects.remove(obs)
        print('removed from data')

print('script done')
```

Figure III. 9: Python Scripting in Blender.

H. Community and Support

Blender is supported by an active global community that contributes to its development, shares tutorials, builds plugins, and provides user support across all skill levels.

III.2.4 Long short-term memory (LSTM)

LSTMs come to the rescue to solve the vanishing gradient problem. It does so by ignoring (forgetting) useless data/information in the network. The LSTM will forget the data if there is no useful information from other inputs (prior sentence words). When new information comes, the network determines which information to be overlooked and which to be remembered [38].

LSTMs have 4 different components, namely

1. Cell state (Memory cell)
2. Forget gate
3. Input gate
4. Output gate

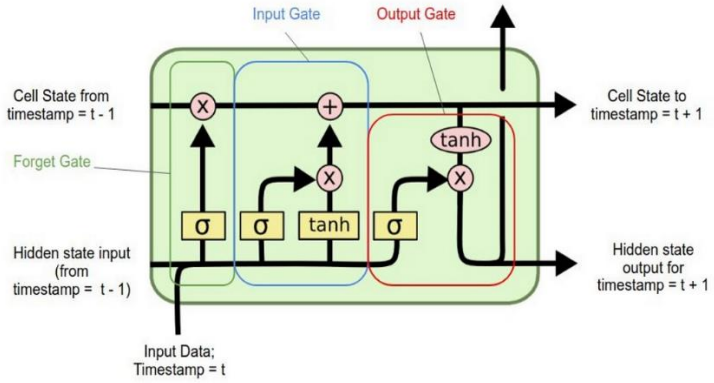


Figure III. 10: LSTM components.

III.2.4.1 Cell state (memory cell)

It is the first component of LSTM which runs through the entire LSTM unit. It kind of can be thought of as a conveyer belt [38].

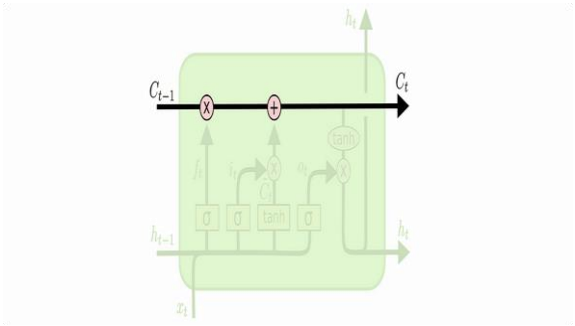


Figure III. 11: LSTM Cell Stat.

This cell state is responsible for remembering and forgetting. This is based on the context of the input. This means that some of the previous information should be remembered while some of them should be forgotten and some of the new information should be added to the memory. The first operation (**X**) is the pointwise operation which is nothing but multiplying the cell state by an array of [-1, 0, 1]. The information multiplied by 0 will be forgotten by the LSTM. Another operation is (+) which is responsible to add some new information to the state [39].

III.2.4.2 Forget gate

The information that is no longer useful in the cell state is removed with the forget gate. Two inputs x_t (input at the particular time) and h_t-1 (previous cell output) are fed to the gate and multiplied with weight matrices followed by the addition of bias. The resultant is passed through an activation function which gives a binary output. If for a particular cell state, the output is 0, the piece of information is forgotten and for output 1, the information is retained for future use [39].

Figure III.12 shows the LSTM Forget Gate. The output f_t is calculated according to Equation (III.1).

$$f_t = \sigma\left(W_f \cdot \left[h_{t-1}, x_t\right] + b_f\right) \quad \text{III.1}$$

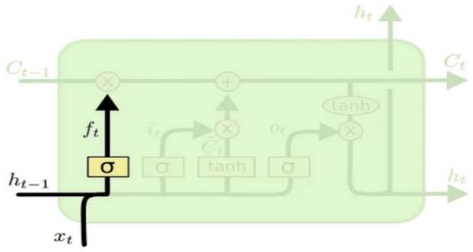


Figure III. 12: LSTM Forget Gate.

III.2.4.3 Input gate

The addition of useful information to the cell state is done by the input gate. First, the information is regulated using the sigmoid function and filter the values to be remembered similar to the forget gate using inputs h_t-1 and x_t. Then, a vector is created using the tanh function that gives an output from -1 to +1, which contains all the possible values from h_t-1 and x_t. At last, the values of the vector and the regulated values are multiplied to obtain useful information [39].

Figure III.13 shows the LSTM forget gate. the output i_t is calculated according to equation (III.2).

$$i_t = \sigma\left(W_i \cdot \left[h_{t-1}, x_t\right] + b_i\right) \quad \text{III.2}$$

Figure III.13 shows the LSTM forget gate. the output $\tilde{c}_t$ is calculated according to equation (III.3).

$$\tilde{C}_t = \tanh\left(W_C \cdot \left[h_{t-1}, x_t\right] + b_C\right) \quad \text{III.3}$$

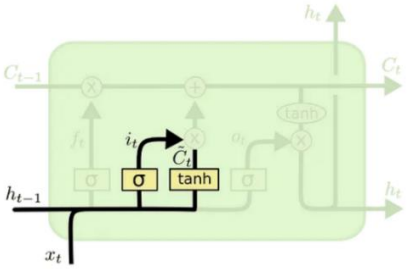


Figure III. 13: LSTM Input gate.

This has 3 parts:

1. A ***sigmoid*** layer decides the values to be updated. This layer is called the "input gate layer"
2. A ***tanh*** activation function layer creates a vector of new candidate values, ***Č(t)***, that could be added to the state.
3. Then we combine these 2 outputs, ***i(t)*** * ***Č(t)*** and update the cell state. The new cell state ***C(t)*** is obtained by adding the output from forget and input gates.

III.2.4.4 Output gate

The task of extracting useful information from the current cell state to be presented as output is done by the output gate. First, a vector is generated by applying the tanh function on the cell. Then, the information is regulated using the sigmoid function and filtered by the values to be remembered using inputs h_t-1 and x_t. At last, the values of the vector and the regulated values are multiplied to be sent as an output and input to the next cell [39].

Figure III.14 shows the LSTM Forget Gate. The output o_t is calculated according to Equation (III.4).

$$o_t = \sigma\left(W_o \cdot \left[h_{t-1}, x_t\right] + b_o\right) \qquad \text{III.4}$$

Figure III.14 shows the LSTM Forget Gate. The output h_t is calculated according to Equation (III.5).

$$h_t = o_t * \tanh\left(C_t\right) \qquad \text{III.5}$$

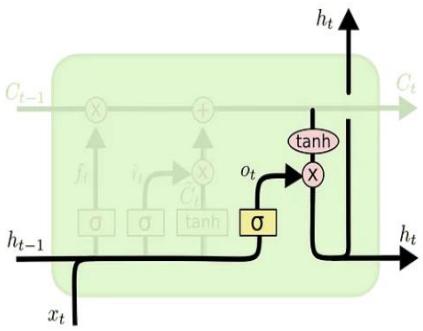


Figure III. 14: LSTM Output gate.

III.2.5 Firebase

Firebase is a cloud-based platform developed by Google that simplifies and accelerates **web and mobile app development.** It offers a comprehensive suite of **backend services, management tools,** and **analytics**, all without the need to manage server infrastructure.

A. Creating a Firebase Project

To begin using Firebase, a project must be created. The process is straightforward :

1. Navigate to https://firebase.google.com and sign in using a Google account.
2. Click **"Get started"**, then select **"Add project"** to begin the project creation process.
3. Assign a name to your project, such as *Incubator DT* (**Figure III.15**).

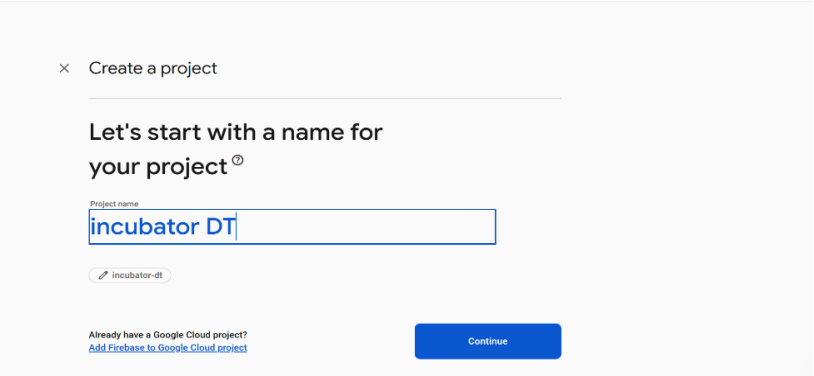


Figure III. 15: Naming the Firebase Project.

4. Disable **Google Analytics** if not required, then click **"Create Project"**.
5. Wait a few moments while Firebase configures the project. Once complete, click **"Continue"**.
6. You will be redirected to your project's console dashboard.

B. Obtaining the API Key

To allow external devices (e.g., ESP32) to communicate with your Firebase project, the API key must be retrieved:

1. In the left navigation pane, go to **Project Settings** see figure III.16**.**

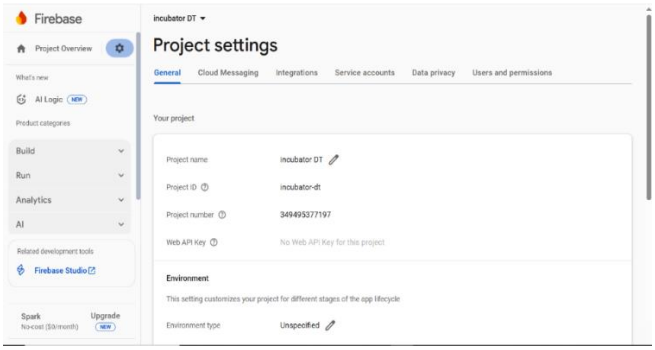


Figure III. 16: Firebase Project Settings.

The API key will be listed under the **General** tab in your project settings.

C. Setting Up the Realtime Database

To store and retrieve data in real time, a Realtime Database must be created and configured:

1. In the left sidebar, select **Realtime Database**, then click **"Create Database"** (figure III.17).

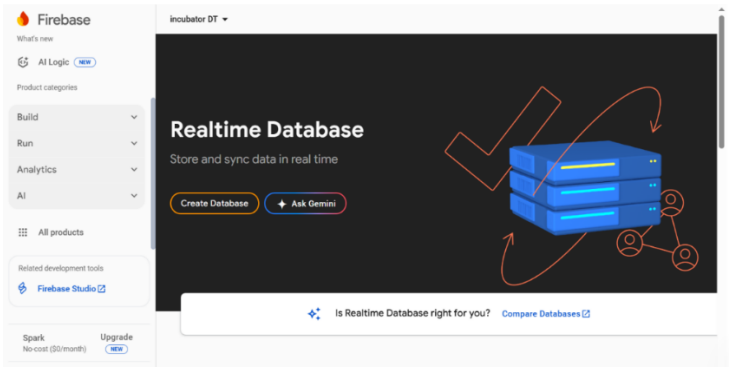


Figure III. 17: Realtime Database Creation Page.

2. Choose a hosting region for your database see figure III.18. This can be based on your geographic location or application requirements.

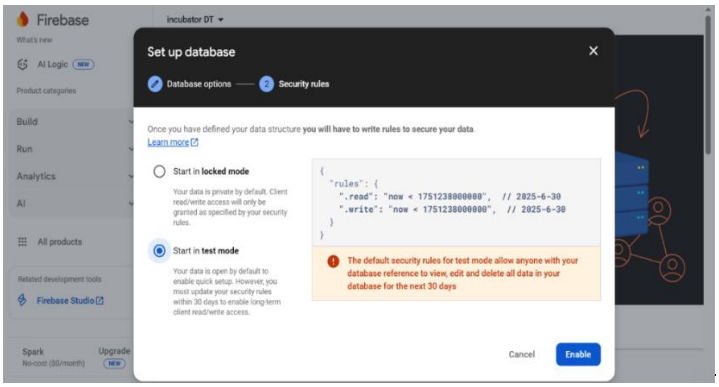


Figure III.18: Selecting Database Region.

3. Set the initial database rules. For development purposes, it is recommended to start in **test mode**, which allows open read/write access temporarily. The database will then be ready for use. The database URL will be provided as shown in figure III.19**.**

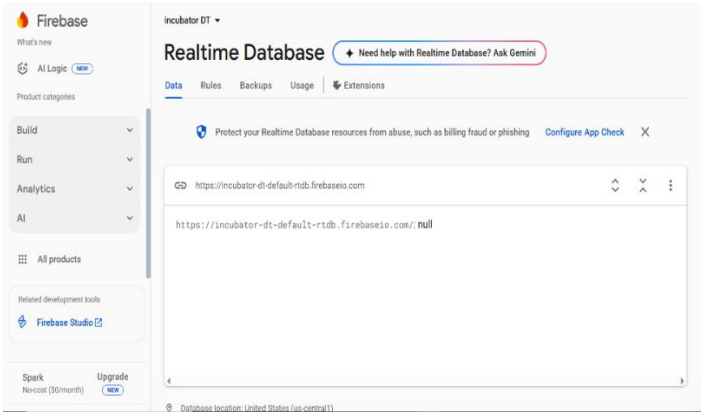


Figure III. 18: Firebase Database URL.

III.2.6 Firestore

A. **Definition of Cloud Firestore**

Cloud Firestore is a **cloud-hosted NoSQL document database** developed by **Google,** designed for building scalable and responsive mobile, web, and server applications. It allows developers to store and synchronize data in **real time** across multiple clients, and supports **offline access**, **complex queries,** and tight integration with other **Firebase services.**

B. Create a Cloud Firestore database.

To set up a Cloud Firestore database within a Firebase project, the following steps must be followed:

1. The Firebase Console must be accessed and the corresponding project opened.
2. In the left-hand navigation panel, the **Build** section should be expanded, and **Firestore Database** selected.
3. The **Create Database** button must be clicked to initiate the database setup (figure III.19).

4. A location for the database must be chosen.
    - If no location selection is available, this indicates that the project's default Google Cloud resource location has already been configured. Several services

within a project may share a common location dependency, which is typically defined during initial project setup or while configuring a related service.

5. The initial security mode for Firestore must be selected.
    - For development purposes, **test mode** is typically used, which allows open read and write access. This mode facilitates testing but should be replaced with stricter rules in production environments. Security configurations can later be adjusted in the **Secure Your Data** section.
6. Finally, the **Create** button must be clicked to complete the database initialization.

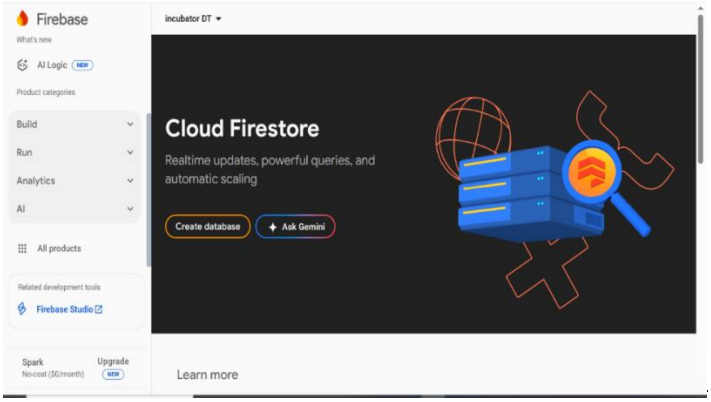


Figure III. 19: Firestore Database Setup Page.

III.3 Hardware tools

III.3.1 ESP-WROOM-32

The ESP-WROOM-32 is a versatile and powerful MCU module that combines Wi-Fi, Bluetooth Classic, and BLE connectivity, making it suitable for a wide range of applications-from low-power sensor networks to demanding tasks like voice encoding, music streaming, and MP3 decoding. At its core lies the **ESP32S** chip, which features two independently controllable CPU cores with adjustable clock speeds ranging from 80 MHz to 240 MHz. The chip also includes a low-power coprocessor that can monitor peripherals while the main CPUs are powered off, enabling highly efficient energy use. The ESP32S integrates a rich set of peripherals, including capacitive touch sensors, a Hall sensor, low-noise amplifiers, SD card support, Ethernet, and multiple high-speed communication interfaces such as SDIO, SPI, UART, and I²C. With Bluetooth, users can connect to mobile devices or broadcast BLE beacons, while Wi-Fi provides long-range communication and direct internet access through routers. Designed for wearable and battery-powered applications, the module consumes less than 5 µA in sleep mode and supports data rates up to 150 Mbps with a maximum output power of 22 dBm, ensuring broad wireless coverage



Figure III. 20: ESP-WROOM-32.

Table III. 1 illustrates the features of the ESP-WROOM-32 module.

Feature	Description
Processor	Dual-core Tensilica LX6 (up to 240 MHz)
RAM	520 KB SRAM
Flash Memory	Up to 4 MB (typical for commercial modules)
Wireless Connectivity	Wi-Fi 802.11 b/g/n and Bluetooth v4.2 (Classic + BLE)
GPIO Pins	30+ configurable pins (ADC, DAC, SPI, I2C, UART, PWM, etc.)
Operating Voltage	3.3V DC
Power Consumption	Low power with Deep Sleep support
Dimensions	Approximately 18 mm × 25.5 mm
Development Platforms	Arduino IDE, PlatformIO, ESP-IDF
Additional Features	Supports encryption (SSL/TLS), OTA updates, and integration into IoT systems

III.3 Conclusion

In this chapter, we reviewed the essential software and hardware tools used in implementing the digital twin system for neonatal incubator management. Python was selected for its powerful capabilities in modeling and predictive analysis, while Visual Studio Code served as the primary environment for effective code development and integration. Blender was used to design the 3D model that digitally represents the incubator. For real-time data storage and synchronization, we relied on the Cloud Firestore database from the Firebase platform. On the hardware side,

the ESP32 microcontroller was employed to collect environmental data from sensors and transmit it wirelessly to the cloud. Together, these tools form a robust and integrated platform that enables accurate real-time monitoring of the incubator and paves the way for developing intelligent models in the upcoming stages of the project.

Chapter IV

Design and Implementation

IV.1 Introduction

The digital revolution is bringing profound changes in the healthcare sector, particularly neonatology. The inclusion of newer technologies such as the Internet of Things (IoT), artificial intelligence (AI), and real-time data analytics has enabled intelligent and adaptive medical systems to be designed. In this regard, the digital twin concept has been a recent innovation offering powerful means of simulation, monitoring, and optimization in the case of neonatal incubators' complex environments.

This chapter describes the design and implementation life cycle of our proposed digital twin-based system for neonatal incubator management. We begin with the system architecture and technology choices that guided our development process. Finally, we describe the deployment, both on the embedded side (ESP32, sensors, data acquisition) and web platform (real-time interface, Firebase integration, 3D visualization), emphasizing the hands-on implementation of our DT.

IV.2 System specifications

In preparing to deploy our smart neonatal incubator management system, based on the digital twin, we developed guidelines that serve as functional and technical requirements for our activity. They help clarify and structure the objectives of this work.

A. Deployment of a data collection and transmission system:
- Include biomedical and environmental sensors in the incubator (temperature, humidity, oxygen level, heart rate, etc.).
- Ensure safe and reliable transmission of collected data to the central platform (Firebase) using the ESP32.

B. IoT-based storage system design:
- Maintain an adequate infrastructure for real-time storage and management of information.
- Provide availability, consistency, and integrity for sensitive medical data.

C. Virtual incubator model development:
- Create a 3D digital representation of the incubator, taking its physical characteristics and newborns' vital parameters into account.
- Develop predictive models to simulate various scenarios for incubator behavior, enabling the forecasting of needs and detection of potential anomalies.

D. Web application:
- Design an intuitive interface in which healthcare professionals can interact with the digital twin, monitor vital signs, and remotely operate the incubator environment.
- Integrate prediction and analysis modules to support real-time medical decision-making.

IV.3 Proposed system

Figure IV.1 illustrates the overall architecture of the proposed intelligent neonatal incubator monitoring system. The proposed system entails several modern technologies, including Internet of Things (IoT), Digital Twin, Firebase, ESP32 microcontroller, and React-based web interface. The proposed system is designed to ensure real-time monitoring and intelligent control of incubator environmental and physiological conditions.

The physical system consists of a neonatal incubator with various sensors connected to an ESP32 board. The sensors track vital parameters such as temperature, humidity, oxygen level, heart rate, respiration, weight, and even vocalization. The ESP32 collects data from the sensors and transfers it to the cloud in a secure manner using WebSocket protocols. At the same time, the board also controls the incubator's actuators, such as the heater, humidifier/ventilator, and alarm systems (sound/light) based on either pre-set conditions or remote commands.

The transmitted data is stored in Firebase using both Cloud Firestore and the Realtime Database. Each sensor or actuator possesses its own organized node in the database, enabling fast, sequential, and real-time data update and fetching. This structure accounts for the efficiency of the system feedback loop.

On the front end, a real-time web application developed using React allows caregivers and clinicians to monitor the incubator, send commands, access the historical database, and interact with the system. The interface fetches data from Firebase and presents it through interactive dashboards, visual alerts, and a 3D model of the incubator. Users can also simulate different scenarios of incubator behavior using embedded predictive models based on LSTM networks for time-series forecasting. This enables better anticipation of system needs and early detection of anomalies.

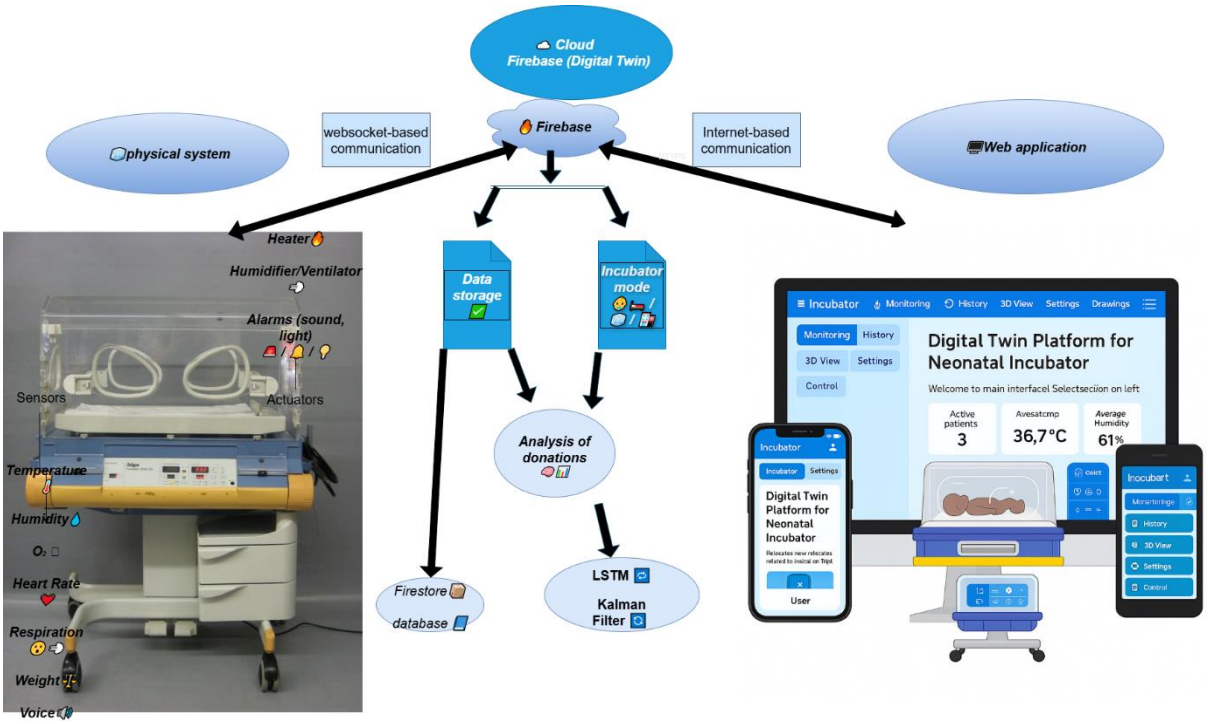


Figure IV. 1: General architecture of the proposed system.

IV.4 Implementation

The proposed Digital Twin-based project is divided into five layers: the physical layer, communication layer, cloud layer, model layer, and web application layer as shown in figure IV.2

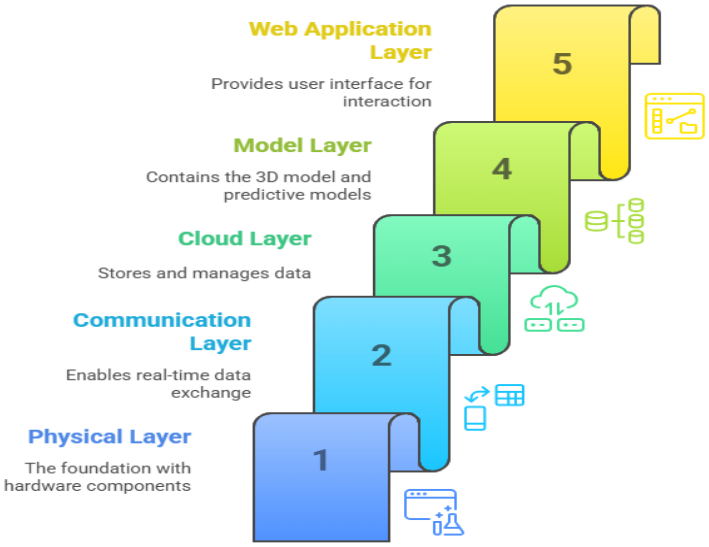


Figure IV. 2: System architecture overview.

IV.4.1 Physical layer

The smart incubator system relies on its lowest physical level which consists of its foundational hardware components. The incubator system consists of a physical device that delivers protected environmental conditions essential for newborns who require premature or medical care. Essential conditions for infant development and well-being include temperature, humidity, and oxygen concentration which this incubator maintains. The incubator incorporates multiple sensors which track environmental parameters continuously to achieve precise real-time data collection. The ESP32 microcontroller at this layer serves as the main control system by linking with sensors and executing preliminary data analysis before transferring information to the cloud. The system enables environmental adjustments through actuators that include heating elements, humidifiers, and oxygen control valves which respond to sensor feedback and system commands from upper levels. The incubator functions as the central component in this layer but its operation depends on the integrated sensors and actuators and ESP32 which together create automatic environmental monitoring and response functions in the physical layer that links the infant's real-world conditions with the digital system see figure IV.3.



Figure IV. 3: Physical Incubator.

IV.4.2 Communication layer

For achieving the embedded part of the system, we used ESP32 board because of its ability to support multiple analog and digital sensors and compatibility with cloud services such as Firebase.

Setup Steps

- **Installing the development environment**

We used the Arduino IDE, which we modified to make it ESP32 compatible.

- **Integrating necessary libraries**

As shown figure IV. 4, we included the following libraries:

- `WiFi.h` for wireless network connection using the ESP32;
- `Firebase_ESP_Client.h` for connection to the Firebase database.

```
EX3.ino
1  #include <WiFi.h>
2  #include <Firebase_ESP_Client.h>
3
```

Figure IV. 4: part of the arduino program.

- **Configuring network credentials**

Wi-Fi SSID and password are defined in the code to create the connection refer to figure IV. 5.

```
// WiFi credentials
const char* ssid = "YOUR_WIFI_SSID";
const char* password = "YOUR_WIFI_PASSWORD";
```

Figure IV. 5: Configuring wi-Fi on Arduino.

- **Firebase Connecting**

We entered the API key, Real-time database address, and Project ID for Firebase (see figure IV. 6).

```
// Firebase credentials
#define API_KEY "YOUR_FIREBASE_API_KEY"
#define DATABASE_URL "https://your-project-id.firebaseio.com/"
```

Figure IV. 6: Connecting ESP32 to Firebase.

IV.4.3 Cloud layer

For our neonatal incubator management project based on a digital twin platform, we have integrated Firebase technology, an IoT platform. It is a layer comprising several subcomponents:

- **Firebase project configuration**

We created a project on the Firebase Console to hold sensor data from the sensors that are inserted into the incubator., The project is designed to facilitate secure, real-time data exchange between the ESP32 microcontroller, prediction algorithms, and user interface.

- **Firebase integration into our application**

We incorporated Firebase libraries in our code, both on:

Arduino IDE: to allow the ESP32 to send sensor data (temperature, humidity, oxygen, heart rate, etc.) to Firebase.

React JS (Web Interface): to show real-time data, set parameters, and visualize predictions in the 3D view.

- **Data storage and structure**

We use two complementary Firebase services:

- Realtime Database: For real-time streaming of data from the hardware. This allows us to view changes in incubator status (alarms, temperature, etc.) in real-time.
- Cloud Firestore: To save historic data, AI model output, and custom configuration parameters for each incubator (e.g., alarm parameters, user preferences, etc.).

IV.4.4 Prediction model layer

IV.4.4.1 3D Model of the incubator

The 3D model was developed using the Blender software, a powerful open-source 3D creation package.

Design process

Design process involved developing an exact digital twin that models the physical incubator from the real world. To meet this, the following Steps for Creating a 3D Model of a Neonatal Incubator Using Blender. The 3D model of the medical incubator was built step by step in Blender using basic geometric shapes and specific modifications for each component, following a logical and practical order:

1. Lower Base (Structural Foundation):

A tensioned cube was used to create a solid and stable foundation. This base provides structural support for all other components of the incubator. The figure IV.7 below illustrates this.

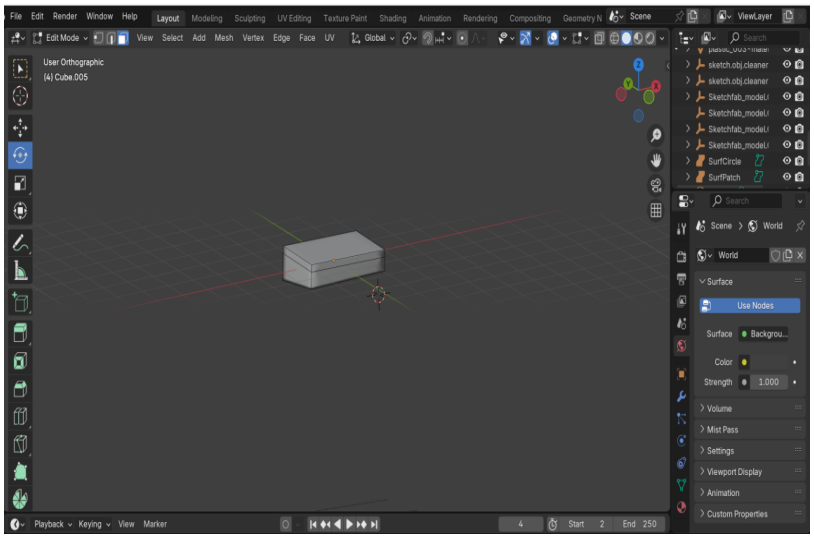


Figure IV. 7: Lower Base.

2. Upper Base:

Another cube, measured along the X and Y axes with a thinner thickness, was placed on top of the lower base to form a platform for the bed and dome.

3. Side handles (Curved handles):

Cylinders were extended horizontally and bent, then placed on the sides of the incubator. These handles mimic the hand grips used to move the device. An example of this design is shown in the figure IV.8 below.

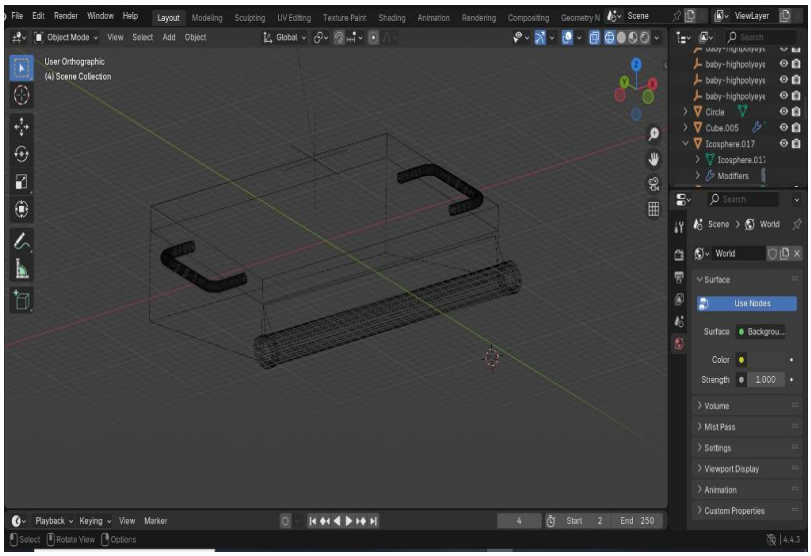


Figure IV. 8: Curved handles.

4. Curved legs:

Vertical cylinders with slight curves were placed at the corners of the base to raise the incubator and provide stable support. The figure IV.9 below illustrates this component.

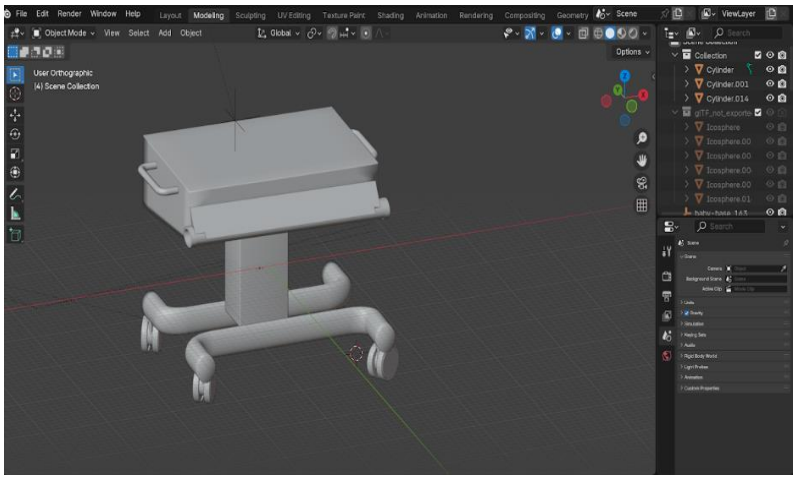


Figure IV. 9: Curved legs.

5. Wheels:

Shorty hemispheres or cylinders were added under the legs, in appropriate sizes, and coated with a metallic material to simulate realistic movement.

6. Drawers (one side only):

Small cubes were added in appropriate sizes and arranged vertically on one side of the base to represent real storage compartments used in medical incubators.

7. Infant Bed (inner):

A resized cube with smooth edges was placed inside the dome, forming a comfortable and safe resting area for the infant.

8. Transparent dome:

A hemisphere was used for ultraviolet light, with its bottom removed and a transparent material applied. This dome symbolizes the visible protective covering covering the infant. The figure IV.10 below clearly illustrates this part.

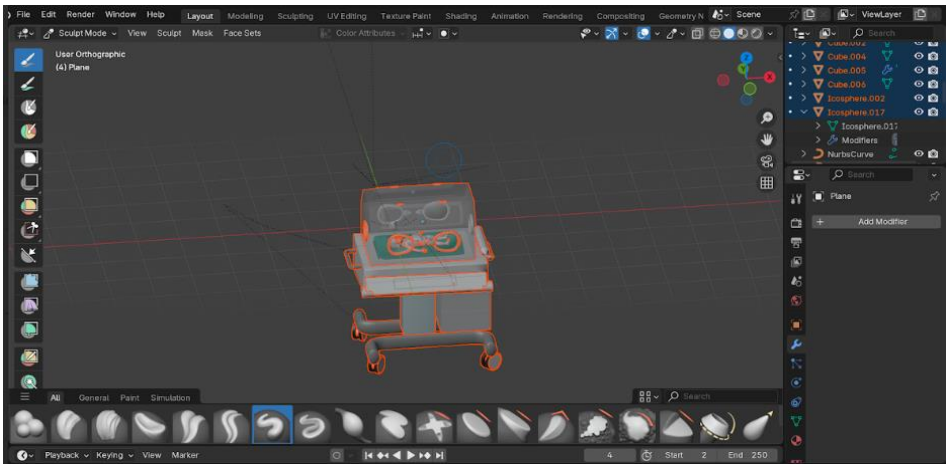


Figure IV. 10: Transparent dome.

9. Screen and Buttons:

A flat cube serves as the screen, while the small cylinders represent control buttons. An image texture was applied to simulate a real digital interface on the screen

10. The Infant and the "Spirit" (Symbolic Element):

A pre-made 3D model of the infant was imported, carefully scaled, and positioned. A glowing translucent ball was added to represent the infant's spirit, adding emotional depth to the scene. This symbolic touch is demonstrated in the following figure IV.11.



Figure IV. 11: Symbolic element.

11. File export:

The model was exported as .glb file as the final file to embed in our web platform for interactive visualization and data overlay.

This 3D model shown in figure IV.12 is used as a graphical representation of the incubator within the digital twin platform, where users can look around and see real-time or projected data within the 3D visualization itself.



Figure IV. 12: Model of the Incubator.

IV.4.4.2 Predictive models

This section provides details about the development of predictive models designed to simulate and predict the behavior of the incubator. Key parameters such as temperature, humidity, carbon dioxide (CO_2) concentration, and infant response (comfort/stress level) have been considered. Several models were developed and compared.

- **Dataset**

  In this project, a dataset was simulated using Python scripts to represent various operational scenarios of the incubator. The simulation covered a one-month period, with samples generated every 5 minutes.

A. Therma dataset

The data collection system, called the "Temperature Dataset," monitors the continuously changing temperature distribution within neonatal incubators. The dataset includes temporal records of incubator and ambient temperature, showing internal and external thermal conditions. The behavior of the incubator control system is captured by the "Heater Status," "Fan Status," and "Door Status" variables. Air movement within the chamber is altered by the "Air Flow Rate," which changes temperature and humidity levels. The dataset contains physiological data, including skin temperature and weight data, to assess the infant's thermal stability. The predictive temperature model generates future estimates of incubator temperature through the "Expected Temperature" column, and the "Risk of Heat Stress" variable identifies risk levels for infants, ranging from low to moderate to high. The dataset serves as training data for machine learning models that create simulations to monitor and optimize incubator thermal environments to ensure infant comfort and safety. The figure IV.13 shows a portion of the text from the thermal dataset.

```
26   df = pd.DataFrame(data)
27   df["predicted_temp"] = df["incubator_temp"] + np.random.normal(0, 0.2, len(timestamps))  # Simulated predictions
28   df.to_excel("thermal_model_dataset.xlsx", index=False)
29   Thermal.py
30   Displaying Thermal.py.
```

Figure IV. 13: Part of the thermal dataset script.

B. Humidity dataset

The dataset logs time-based information about incubator_humidity and ambient_humidity parameters that monitor humidity levels inside and outside the incubator. The humidifier_status and airflow_rate track how the incubator's humidity control system operates to control moisture distribution and air movement patterns. The environmental elements door_status and fan_status create changes in humidity consistency through their ability to bring external air inside or enhance evaporation processes. The dataset contains skin_temp together with weight_g and respiratory_rate as physiological indicators that show how humidity affects infant comfort and respiratory health. Moreover, the dataset combines actual sensor data with forecasted humidity values in the predicted_humidity column and risk level classification into Low, Moderate, or, high risk categories.

To generate the predicted humidity values, Gaussian noise is added to the original humidity readings, simulating forecasting uncertainty. This is illustrated in figure IV.14, which shows the

code used to create the predicted_humidity feature and export the dataset to an Excel file. This process enhances the dataset with simulated predictions, supporting machine learning model development for humidity control in neonatal care environments.

```
28  df = pd.DataFrame(data)
29  df["predicted_humidity"] = df["humidity"] + np.random.normal(0, 1, len(timestamps))
30  df.to_excel("humidity_model_dataset.xlsx", index=False)
```

Figure IV. 14: Part of the humidity dataset script.

C. CO_2 concentration dataset

The CO_2 concentration dataset captures the dynamics of carbon dioxide levels within the neonatal incubator environment, which play a critical role in assessing and ensuring optimal air quality for newborns. The dataset includes time-stamped records of co2_level and airflow_rate, reflecting the buildup and dilution of carbon dioxide due to ventilation efficiency. The oxygen_valve_status and door_status parameters provide insight into how external interventions or environmental access points affect gas exchange within the incubator.

Figure IV.15 illustrates a snippet of code used to generate part of this dataset, particularly the simulated predictions of oxygen concentration based on existing O_2 readings

```
29  df = pd.DataFrame(data)
30  df["predicted_O2"] = df["O2_level"] + np.random.normal(0, 1, len(timestamps))  # Simulated predictions
31  df.to_excel("oxygen_model_dataset.xlsx", index=False)
```

Figure IV. 15: Part of the CO_2 dataset script.

Additionally, the dataset incorporates fan_status to indicate mechanical support in regulating air circulation. From a physiological standpoint, it includes skin_temp, weight_g, and respiratory_rate as indicators of the infant's metabolic activity and response to CO_2 fluctuations. The dataset also features predicted_co2_level, simulating model-driven forecasting based on environmental and physiological variables, and a co2_risk_level classification (Low, Moderate, High) that helps in early detection of potential respiratory distress caused by elevated CO_2 exposure.

By integrating sensor data with predictive analytics and risk stratification, this dataset supports the development of intelligent incubator systems capable of maintaining ideal air quality and enhancing neonatal safety.

D. Baby response dataset

The baby response dataset captures the complex interplay between environmental stimuli and neonatal physiological responses, aiming to evaluate and predict the comfort or stress levels of newborns in real time. The dataset logs time-based measurements of incubator_temp, noise_level, and light_intensity, which represent the thermal, auditory, and visual stimuli in the incubator environment.

The dataset also includes mattress_status and movement, reflecting the physical conditions and activity level of the infant. Critical physiological data are captured through heart_rate, crying, and apnea_risk, offering insight into the baby's autonomic responses and possible signs of discomfort or respiratory distress.

To synthesize overall well-being, the dataset features a baby_comfort_index as an aggregated metric that reflects the infant's general comfort state. Furthermore, a predicted_response column is introduced, which represents forecasted behavioral or physiological reactions based on current environmental and biological inputs.

To generate the predicted_response, a predictive model simulates future baby responses using variables such as heart rate variability, crying detection patterns, and external conditions. This is illustrated in figure IV.16, which presents a sample code snippet used to create the predicted_response field and export the data for model development and early warning system integration

By combining environmental, behavioral, and physiological indicators with predictive modeling, this dataset supports the development of smart monitoring systems for neonatal care. These systems can provide early alerts to caregivers, improving reaction time and ultimately enhancing infant comfort and safety within the incubator.

```
28    df = pd.DataFrame(data)
29    df.to_excel("baby_interaction_dataset.xlsx", index=False)
```

Figure IV. 16: Part of the baby response dataset script.

Finally, each dataset contains 30 days of data recorded at 5-minute intervals, producing high-resolution time series suitable for deep learning applications.

These predictive models are integrated into the system to simulate real-time conditions, predict future risks, and enhance proactive control of the neonatal incubator environment.

- **Performance Criteria**

  To evaluate the developed models, the following performance metrics were considered:

  - RMSE (Root Mean Square Error): Measures the average magnitude of the prediction errors.
  - R^2 (Coefficient of Determination): Indicates how well the model explains the variance in the data.
  - Accuracy: Represents the proportion of correct predictions among the total number of cases evaluated.

IV.4.4.2 LSTM architectures

In order to empirically study systematically the performance and appropriateness of deep learning techniques, i.e., recurrent neural networks with the Long Short-Term Memory (LSTM) architecture, in predictive modeling in neonatal incubator environments, we conducted an empirical study that involved the implementation and benchmarking of four varied LSTM model settings. The main goal of this evaluation was to determine the best model structure for each targeted variable category, for example, temperature, humidity, CO_2 level, and baby

comfort level. Each model was trained and tested from time-series datasets specially tuned to the problem domains.

The four LSTM models under evaluation in this study are described below:

A. Baseline LSTM model

The Baseline LSTM model is the most basic implementation of the LSTM architecture. It involves a single LSTM layer constructed with a fixed amount of hidden units (neurons), and a Dense (fully connected) output layer responsible for delivering the end-predicted value.

This model is mostly utilized as a benchmark point for gauging the incremental gains of architectural modifications made in subsequent models. The Baseline LSTM model's simplicity helps to establish a baseline point of performance by which all other additional architectural modifications can be quantitatively measured.

Advantages:

- Simplified structure makes it easier to interpret.
- Faster training time and lower computational expense through reduced parameters.

Limitations:

- Ability to learn complex or non-linear temporal dependencies is limited.
- Susceptible to underfitting and overfitting depending on dataset complexity.

B. LSTM + dropout model

The LSTM + dropout model builds on the baseline model by incorporating a Dropout layer immediately following the LSTM layer. The Dropout layer is used to prevent a proportion of LSTM units from being active in each training step, by a specified dropout rate (usually 0.2 to 0.5).

This stochastic deactivation procedure is intended to reduce the problem of overfitting, particularly when dealing with high-dimensional input characteristics or limited training data. By avoiding overreliance on any single neuron, dropout promotes generalization and subsequently improved prediction on unseen (test) data.

Strengths:

- Regularization process improves generalization performance.
- Most beneficial during training data with noise or large variance.

Weaknesses:

- Partially longer training time due to injected noise during learning.
- Risk of underleveraging model potential if dropout is too high.

C. LSTM + dense model

LSTM + dense incorporates an architectural modification by the addition of one or several intermediate dense layers between the LSTM layer and the final output layer. The primary motivation for this design is to increase the model's ability to transform and represent features non-linearly, a requirement where the connection between input features and the target variable is not simple or very non-linear.

The use of Dense layers facilitates hierarchical feature abstraction, enabling the model to learn complex patterns across time steps, particularly when interdependencies exist in multivariate inputs (e.g., sensor reads, environmental conditions).

Advantages:

- Enhances expressiveness of the model and approximation capabilities.
- Can be used in data sets where output is influenced by variable interactions.

Limitations:

- Increased model complexity and likelihood of overfitting.
- Increased training times and memory requirements.

D. Bidirectional LSTM model

Bidirectional LSTM utilizes a bidirectional wrapper around the LSTM layer. This configuration allows the model to propagate temporal data sequences in both directions and therefore learn contextual dependencies not only from the past (as in LSTMs) but also from the future.

Bidirectional LSTMs are particularly useful in scenarios where the output of one specific time step depends on both the prior and subsequent values in the time series a condition that often applies in physiological data streams and environmental control systems for neonatal incubators.

Architecture is implemented by creating two different LSTM layers one processing input from t_0 to t_n, and the other from t_n to t_0 whose outputs are usually concatenated before they reach the output layer.

Advantages:

- Improved performance in extracting long-distance and bi-directional dependencies.
- Strong pattern recognition between sequences.

Drawbacks:

- Computational expensive due to doubled LSTM operations.
- Not always necessary for causal (real-time) prediction where future inputs are not available.

Finally, each of the four LSTM architectures evaluated in this study reflects a specific modeling hypothesis related to temporal dependent learning, generalization control, and nonlinear representation capacity. The Baseline model is the minimally viable implementation, while the Dropout, Dense, and Bidirectional variants seek to improve their limitations through structural optimizations.

In the following sections, we compare the experimental performance of these models using evaluation metrics such as root mean square error (RMSE) and mean absolute error (MAE). Furthermore, graphical illustrations including training loss curves, predicted versus actual graphs, and scatterplots are presented to facilitate visual comparison of learning behavior and prediction accuracy.

1. Temperature prediction results using LSTM variants

Four of LSTM variants were employed to predict temperature in the incubator: Baseline LSTM, LSTM with Dropout, LSTM with Dense, and Bidirectional LSTM. Validation of each of the models was done using composite plots with training/validation loss curves, actual vs. predicted temperature, and scatterplots, which are given below as figures IV.17 to IV.20.

a) Baseline LSTM

The baseline model exhibited a rapid and steady convergence during training. The predicted temperature curve closely matched the actual values with minimal lag during periods of transition. The scatterplot presented good alignment between predicted and true values, indicative of a correct baseline performance The baseline model achieved performance with: RMSE = 0.0896, R^2 = 0.9808, and Accuracy = 100%.

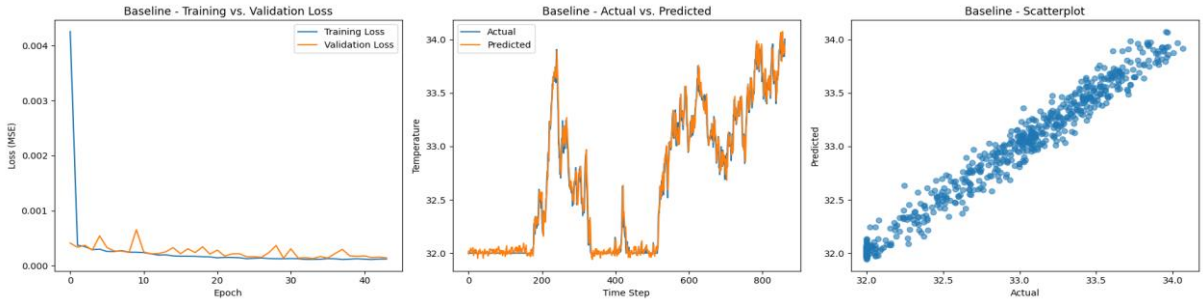


Figure IV. 17: Baseline LSTM results.

b) LSTM with dropout

Adding dropout introduced variance to the validation loss. This was a sign that the model was trying to reduce overfitting. The forecasted temperature had a lag in reacting to steeply changing inputs. The points of the scatterplot were more dispersed than the baseline, suggesting lower correlation. The dropout model achieved performance with: RMSE = 0.153372, R^2 = 0.943744, Accuracy = 1.0.

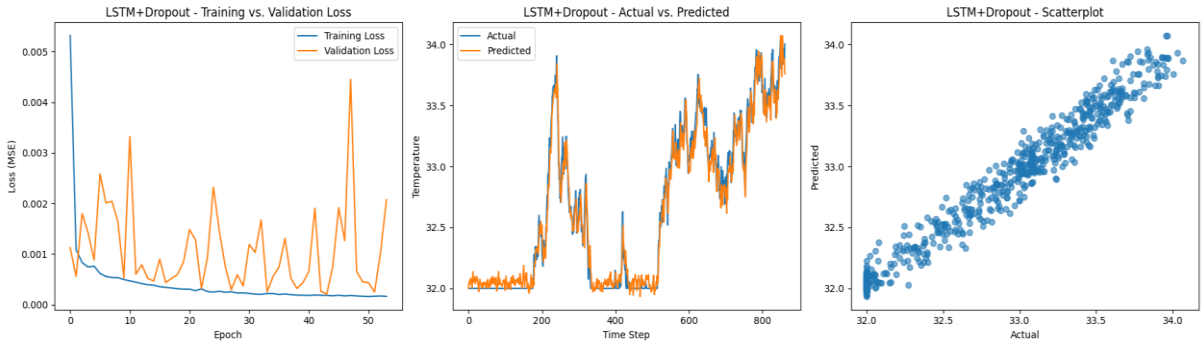


Figure IV. 18: LSTM with Dropout results.

c) LSTM with dense layer

This model provided smoother validation curves compared to the dropout version. The temperature curve it forecasted closely tracked the observed data, especially at steady conditions. The scatterplot became denser along the diagonal line, which indicated that prediction consistency was more robust. The dense model achieved performance with: RMSE = 0.118161, R^2 = 0.966609, Accuracy = 1.0.

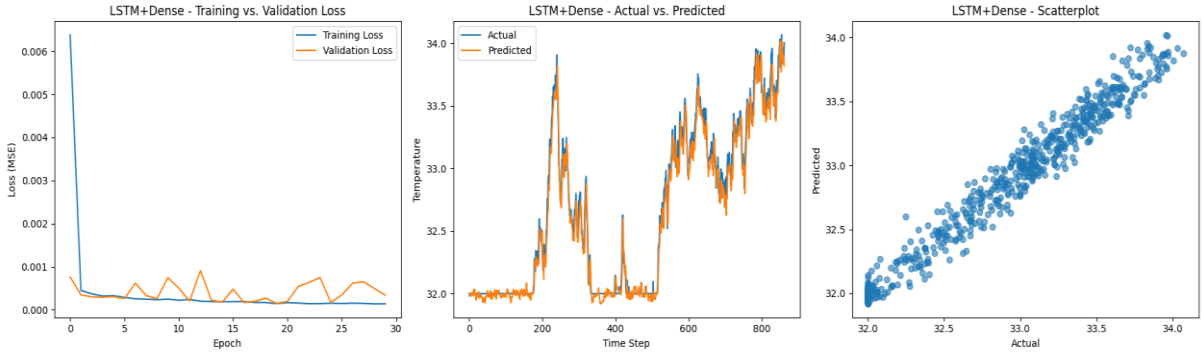


Figure IV. 19: LSTM + Dense results.

d) **Bidirectional LSTM**

This model worked the best. Loss curves were extremely stable, and the calculated temperature curve perfectly overlaid with the real values for all time steps. The scatterplot showed tightly packed points along the ideal diagonal, which confirms the high accuracy and stability of the model. The bidirectional model achieved performance with: RMSE = 0.088157, R^2 = 0.981414, Accuracy = 1.0.

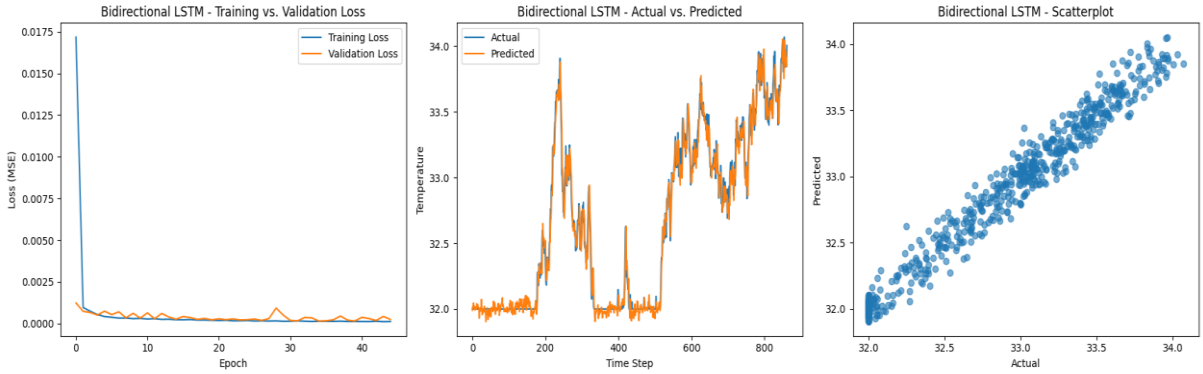


Figure IV. 20: Bidirectional LSTM results.

Table IV.1 presents a comprehensive comparison of the four LSTM variants used for incubator temperature prediction, based on both quantitative performance metrics and qualitative observations.

The Bidirectional LSTM model yielded the highest overall performance of the variants, demonstrated through the lowest RMSE and highest R^2 which indicated the best reliability and predictive accuracy. The Baseline LSTM also had solid predictive performance, but having an additional Dense layer permitted for slightly more consistent results. The Dropout variation had the lowest performance of the models, even though it offered regularization. The Bidirectional LSTM model should be a logical recommendation for predictive modeling of temperature environments in incubator settings.

Table IV. 1 LSTM Model performance for predicting incubator temperature

Model	RMSE	R^2	Accuracy	Key Observations
Baseline LSTM	0.089559	0.980818	1.0	Strong performance with fast convergence. Predictions closely match actual values.
LSTM with Dropout	0.153372	0.943744	1.0	Higher variance in validation loss. Slower reaction to sudden changes. More scattered predictions.
LSTM with Dense Layer	0.118161	0.966609	1.0	More stable than Dropout model. Performs well in stable conditions. Less scattered results.
Bidirectional LSTM	0.088157	0.981414	1.0	**Best performance overall**. Very stable loss curves and predictions almost perfectly align with actual values.

2. Humidity forecasting results using LSTM variants

As with temperature prediction, four variants were compared for humidity prediction. Model validation for each was performed using composite plots training/validation loss curves, actual vs. predicted humidity traces, and scatterplots shown in figures IV.21 to IV.24.

Baseline LSTM

The baseline model converged rapidly and reached a stable training/validation-loss plateau near 0.063.

Its forecast-humidity curve shadowed the observed values in steady intervals, but under-shooted and lagged sudden changes.

The scatterplot showed broad scattering of points from the diagonal, showing poor correlation. The baseline model achieved performance with: RMSE = 6.030862 ; R^2 = 0.002567 Accuracy = 0.857309.

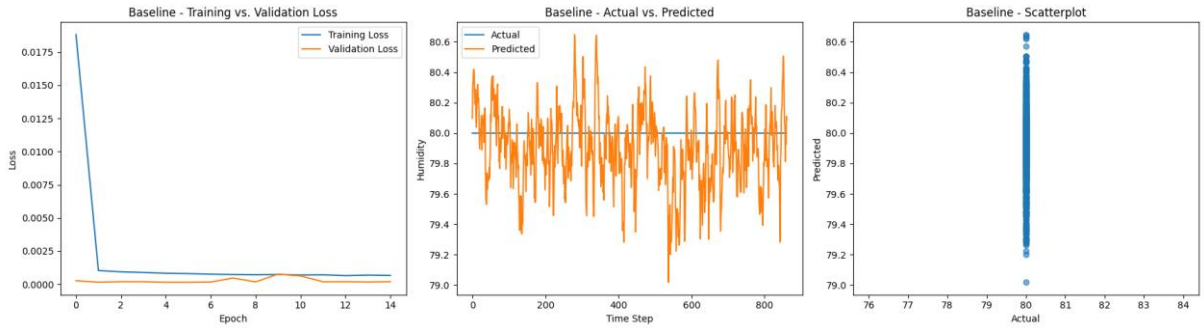


Figure IV. 21: Baseline LSTM results.

a) LSTM with Dropout

Including a dropout layer introduced greater variability in the validation-loss curve, typical of regularization.

Predictions reacted somewhat slower to abrupt humidity changes and the scatterplot was somewhat wider than control. The dropout model achieved performance with: RMSE = 6.030842 ;R^2 = 0.002574; Accuracy = 0.857309.

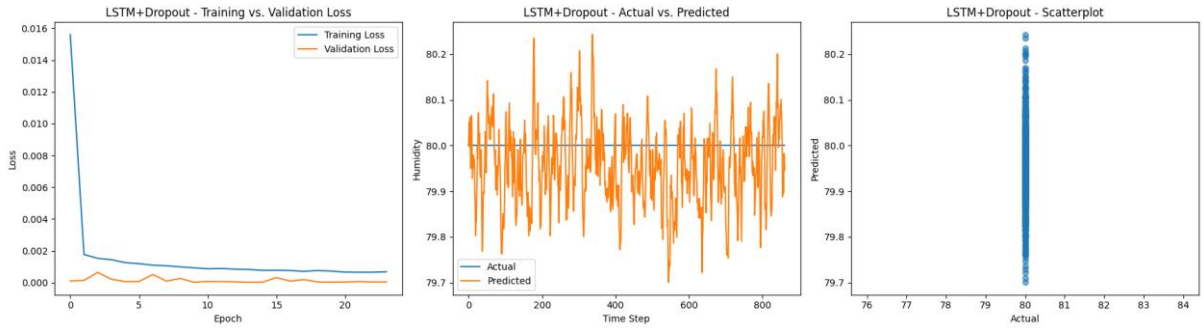


Figure IV. 22: LSTM with Dropout results.

b) LSTM with Dense Layer

The addition of a second dense layer made training and validation losses smooth, even if they did not decrease their final value.

Predicted humidity approached actual measurement when incubator conditions were stable. Scatter points became tighter on the best line. The dense model achieved performance with: RMSE = 6.037980 ; R^2 = 0.000211 ; Accuracy = 0.857309.

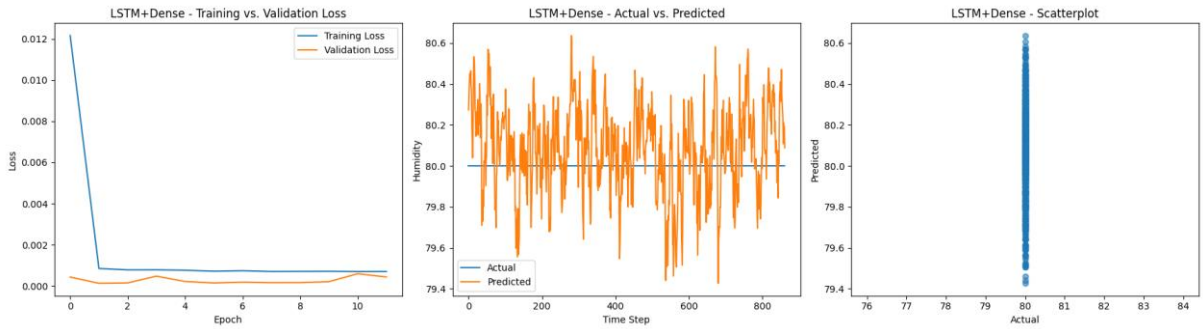


Figure IV. 23: LSTM + Dense results.

c) Bidirectional LSTM

This model generated the most stable loss curves of the four.

Its output tracked the true humidity curve closer than the other models, but there were still large errors at abrupt transitions.

Scatterplot points tended to cluster closer to the diagonal, indicating a modest improvement in correlation. The bidirectional model achieved performance with: RMSE = 6.036652 R^2 = 0.000651 ; Accuracy = 0.857309.

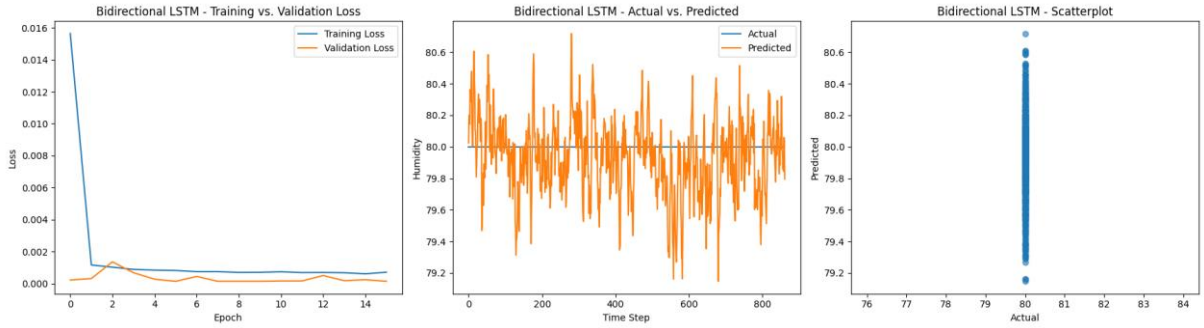


Figure IV. 24: Bidirectional LSTM results.

Table IV.2 presents a comprehensive comparison of the four LSTM variants used for incubator humidity prediction, based on both quantitative performance metrics and qualitative observations.

Table IV. 2 LSTM Model performance for predicting incubator humidity

Model	**RMSE**	**R^2**	**Accuracy**	**Key Observations**
Baseline LSTM	6.030862	0.002567	0.857309	Fast convergence; reasonable tracking in steady zones; poor response to sudden changes.
LSTM + Dropout	6.030842	0.002574	0.857309	More variance in validation loss; slightly slower to react; more dispersed predictions.
LSTM + Dense	6.037980	0.000211	0.857309	Smoother loss curves; slightly better in stable humidity; no significant overall gain.
Bidirectional LSTM	6.036652	0.000651	0.857309	Most stable training; slightly tighter fit; large errors remain in rapid transitions.

While the Bidirectional LSTM had the smoothest training dynamics, none of the variants showed strong statistical performance each had large RMSE values (\~6 % RH) and were close to zero in R^2, indicating they were not capturing much of the underlying humidity dynamics. Additional enhancements will most likely need to come from richer input features, more data or alternative model architectures before failure-proof incubator humidity control can be accomplished

3. Oxygen-concentration prediction result using LSTM variants

As with previous predictions, four Oxygen prediction variables were compared. The model was validated for each using composite plots training/validation loss curves, actual versus predicted humidity effects, and scatterplots shown in figures IV.25 to IV.28.

a) Baseline LSTM

The baseline network experienced a flat loss plateau at a 0.040 – 0.042 value for the first 20 epochs.

Its predicted-oxygen profile followed the observed one through smooth variation but deviated from it when changes were abrupt.

The scatter-plot exhibited much spread and, therefore, weak correlation. The baseline model achieved performance with RMSE = 2.099195. R^2 = – 0.014708. Accuracy = 1.0.

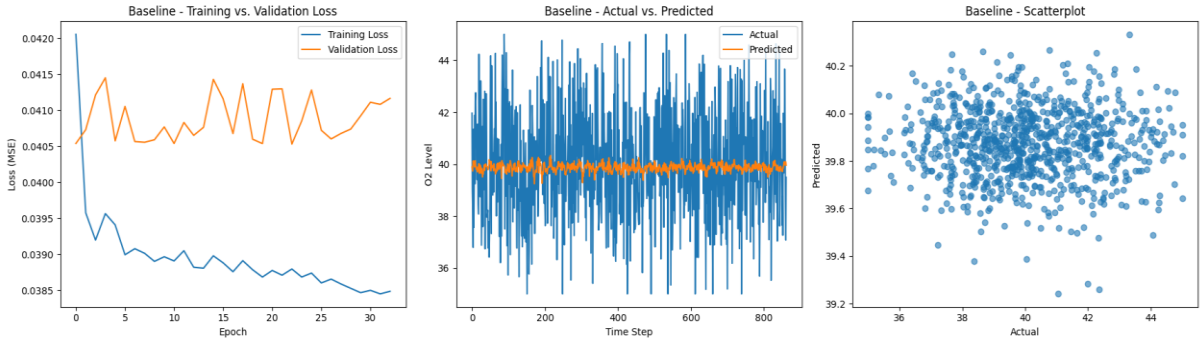


Figure IV. 25: Baseline LSTM results.

b) LSTM with Dropout

Adding a dropout layer introduced more oscillations in the validation-loss graph, a sign of regularization under way.

The model's responses to abrupt oxygen changes were marginally slower than baseline, and the points of scatter were marginally more dispersed. The dropout l model achieved performance with : RMSE = 2.087905 R^2 = –0.003823 Accuracy = 1.0.

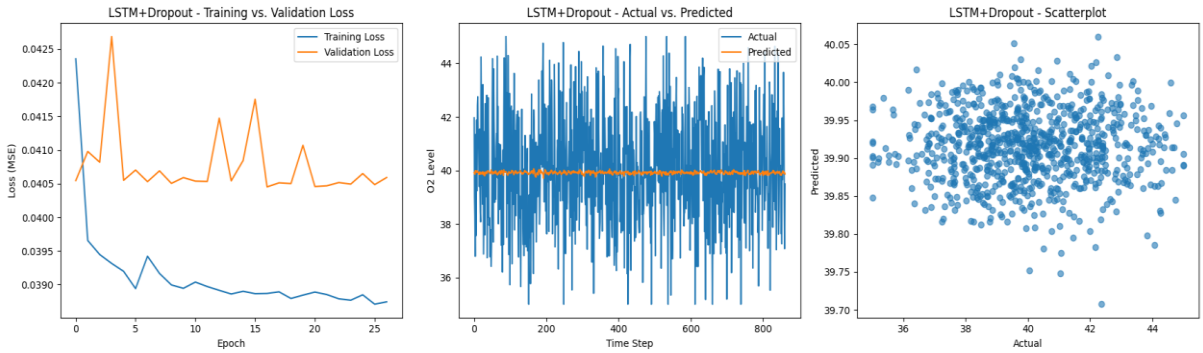


Figure IV. 26: LSTM with Dropout results.

c) LSTM with Dense Layer

Insertion of another dense layer smoothened training and validation loss without lowering their ultimate values.

Predicted-oxygen trace was a little better correlated with actual readings in steady periods, and density of scatter was improved on the diagonal. The dense model achieved performance with: RMSE = 2.094604. R^2 = –0.010275. Accuracy = 1.0

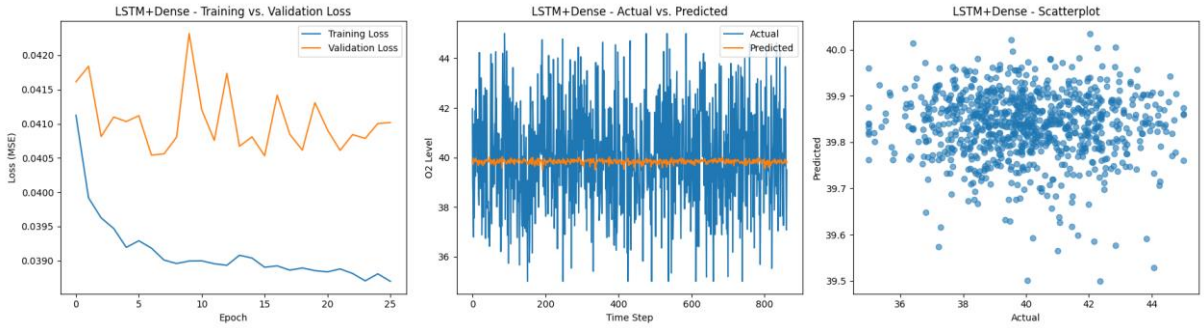


Figure IV. 27: LSTM + Dense results.

d) Bidirectional LSTM

The bidirectional one generated the most flattened loss curves out of the four.

Its prediction tracked the true oxygen series close to the dense model but still retained visible errors at steep transitions.

Scatter-plot points were very slightly tighter than the baseline. The bidirectional model achieved performance with: RMSE = 2.100185. R^2 = –0.015666 Accuracy = 1.0

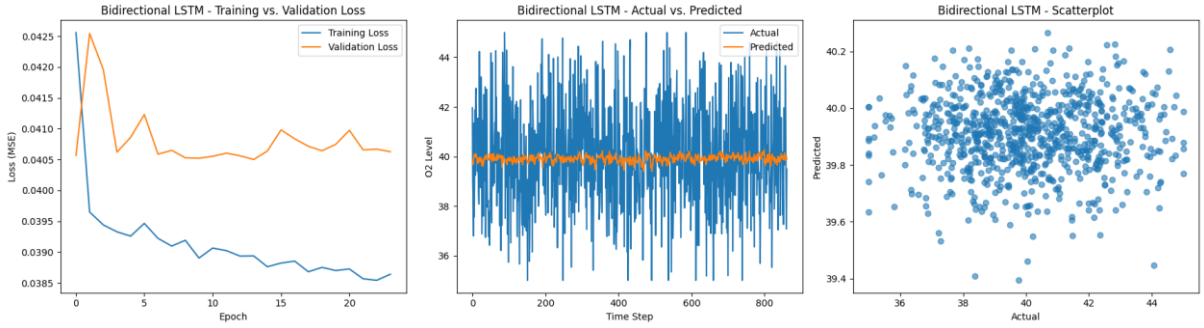


Figure IV. 28: Bidirectional LSTM results.

Table IV.3 presents a comprehensive comparison of the four LSTM variants used for incubator oxygen prediction, based on both quantitative performance metrics and qualitative observations.

While the LSTM with Dropout achieved the lowest RMSE (2.087905) and best R^2 (–0.003823, nevertheless negative), all four of the variants produced very similar and ultimately weak statistical results:

Large absolute errors (\~ \u0026 2 % O_2), and negative R^2 indicates that none of the models represented adequately the oxygen dynamics.

All methods achieve 100% accuracy and 1×1 confusion matrices when the target is highly imbalanced or single-class, hence it is not an informative measure in this scenario.

Recommendation: richer input features (e.g., CO 2 level, fan speed), a wider oxygen operating range, or alternative model architectures would likely be required before reliable oxygen-level prediction can be achieved for incubator control

Table IV. 3 LSTM Model performance for predicting incubator oxygen-concentration

Model	RMSE	R^2	Accuracy	Key Observations
Baseline LSTM	2.099195	–0.014708	1.0	Fast convergence; decent tracking during gradual changes; weak response to sudden shifts.
LSTM + Dropout	2.087905	–0.003823	1.0	Higher variance in loss; slower adaptation to oxygen spikes; slightly noisier predictions.
LSTM + Dense Layer	2.094604	–0.010275	1.0	Smoother training curve; better fit in stable periods; overall performance close to baseline.
Bidirectional LSTM	2.100185	–0.015666	1.0	Most stable training; minor visual improvement in scatterplot; no real gain in error metrics.

4. Baby-Response prediction results using LSTM variants

As with previous predictions, four baby-Response prediction variables were compared. The model was validated for each using composite plots training/validation loss curves, actual versus predicted humidity effects, and scatterplots shown in Figures IV.29 to IV.32.

a) Baseline LSTM

The baseline network converged well, with training and validation loss plateauing between 0.038 – 0.041 after around 30 epochs.

Its predicted-response plot tracked measured values for slow changes but diverged for more sudden transitions.

Its scatter-plot showed a large spread around the diagonal, reflecting poor correlation. The baseline l model achieved performance with: RMSE = 0.119501 R^2 = -0.004808 Accuracy = 1.0

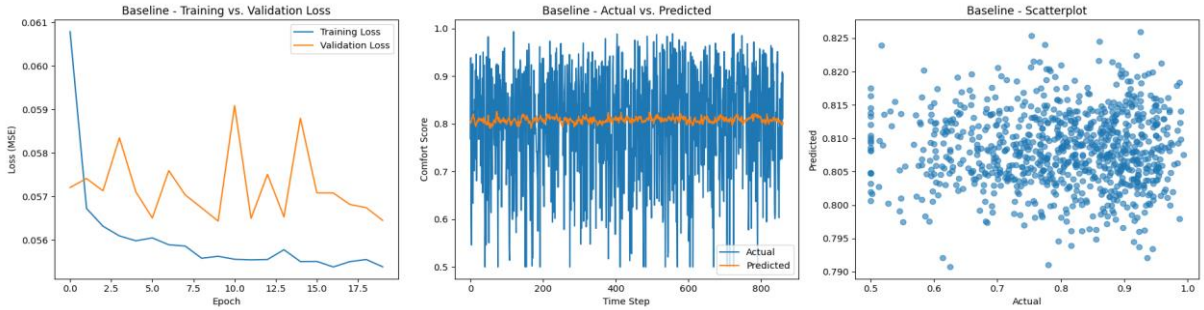


Figure IV. 29: Baseline LSTM results.

b) LSTM with Dropout

Adding a dropout layer introduced noise into the loss curves but yielded the lowest total error. Forecasts reacted to abrupt response shifts a bit better than the baseline, and scatter points were

slightly nearer the ideal line. The dropout model achieved performance with: RMSE = 0.119369 R^2 = -0.002584 Accuracy = 1.0

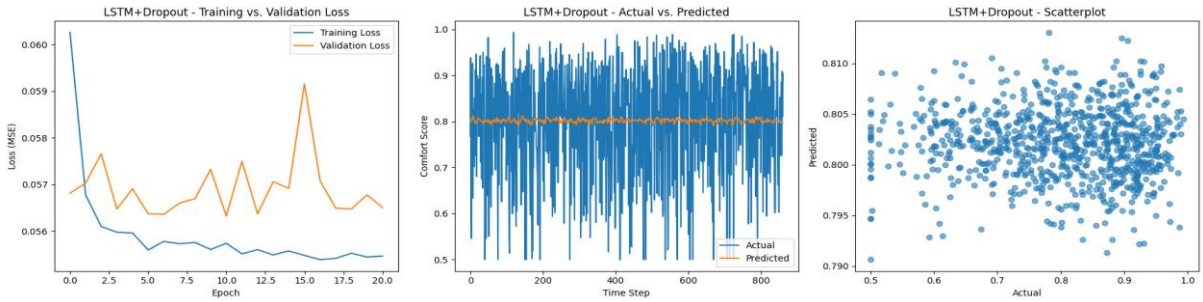


Figure IV. 30: LSTM + Dropout results.

c) LSTM with Dense Layer

The addition of another dense layer smoothed training and validation losses without reducing their final values.

Predicted response adhered to actual readings in constant segments a bit better, and scatter density was better along the diagonal. The dense model achieved performance with: RMSE = 0.119752 R^2 = -0.009033 Accuracy = 1.0

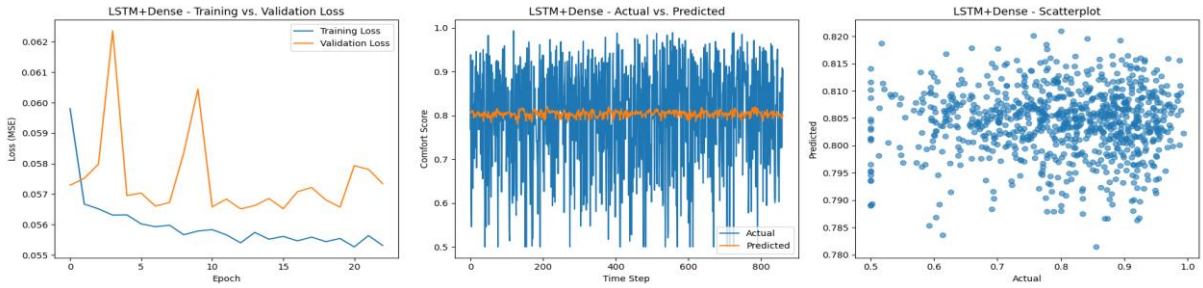


Figure IV. 31: LSTM + Dense results.

d) Bidirectional LSTM

Its bidirectional counterpart produced the flattest loss curves but showed no quantitative improvement over the other models.

Its output traced the true response series in the same manner as the dense model, with evident errors persisting at fast transitions. The bidirectional model achieved performance with: RMSE = 0.119955 R^2 = -0.012450 Accuracy = 1.0

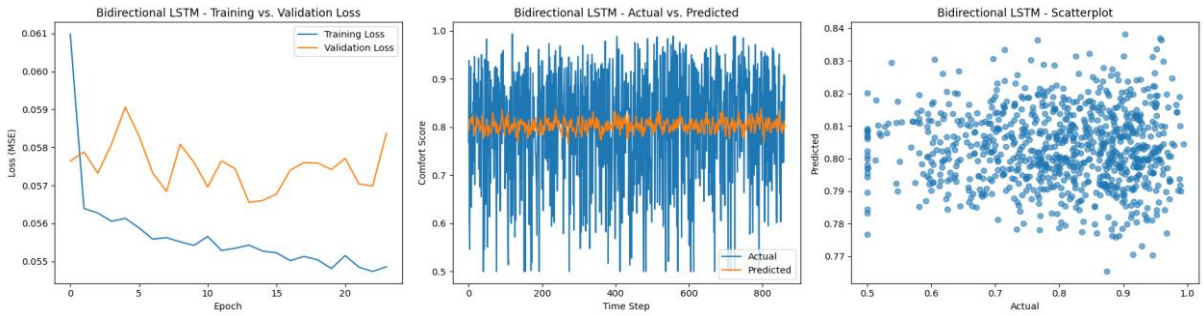


Figure IV. 32: Bidirectional LSTM results.

Table IV.4 presents a comprehensive comparison of the four LSTM variants used for incubator baby-response prediction, based on both quantitative performance metrics and qualitative observations.

Table IV-4:

Table IV. 4LSTM Model performance for predicting incubator baby-response

Model	**RMSE**	**R^2**	**Accuracy**	**Key Observations**
Baseline LSTM	0.119501	-0.004808	1.0	Stable convergence; decent tracking during slow variations; weak on sharp transitions.
LSTM+ Dropout	**0.119369**	**-0.002584**	1.0	Lowest RMSE; slightly more volatile loss; small improvement in capturing transitions.
LSTM + Dense Layer	0.119752	-0.009033	1.0	Smoother training; similar metrics to baseline; slight improvement in steady-state performance.
Bidirectional LSTM	0.119955	-0.012450	1.0	Most stable training process; no significant improvement; slightly tighter scatter.

The LSTM with Dropout did the best numerically (smallest RMSE and least negative R^2), but all four basically performed equally and not very well from a regression perspective:

Absolute errors are still about ±0.12 response-units; and all R^2s, negative, revealing the models all missed the underlying baby response dynamics.

Perfect accuracy and 1 × 1 confusion matrices indicate a single-class nature or extremely imbalanced target, so one cannot utilize accuracy here.

Recommendations: For more reliable estimates to inform one model vs another, better baby-response prediction is expected to require richer input features (such as vital signs or environmental cues), an expanded action-space, or alternative design of model architectures such that decisions are trustworthy to operationalize for feedback control of an incubator.

Across all four prediction tasks temperature, humidity, oxygen concentration and Baby-Response the four LSTM variants demonstrated varying degrees of performance. Bidirectional LSTM consistently emerged as the best performing model in terms of stability and prediction accuracy, particularly in the temperature prediction task, where it achieved the lowest RMSE and highest R^2, with predictions that matched almost perfectly with the actual values. Baseline LSTM also showed strong results, particularly for temperature, indicating that even a simple LSTM structure can be effective under certain conditions. Dense-layer LSTM generally improved training stability and produced denser scatterplots, indicating slightly better prediction consistency under stable conditions, although its statistical gains were modest.

Meanwhile, Dropout LSTM, despite introducing regularization, tended to perform poorly in most cases, likely due to increased validation loss variance and slow response to sudden changes. However, for humidity and oxygen concentration forecasts, all models struggled, with low or even negative R^2 values and relatively high root mean square (RMSE) coefficients, suggesting that LSTM structures alone may not be sufficient to capture the complex dynamics of these variables. These results suggest that while LSTM models, particularly bidirectional ones, hold promise for temperature modeling, improving humidity and oxygen forecasts may require enhanced input features, hybrid models, or alternative structures.

Regarding the negative values, I will explain something about that. The negative R^2 values recorded in some models, particularly those related to humidity and oxygen forecasts, can be attributed to limited variance in the target data, the absence of important explanatory features, and the unbalanced and noisy nature of the dataset. Together, these factors hindered the model's ability to learn accurate representations of system dynamics, indicating a need to improve data quality, diversity, and feature engineering in future work.

IV.4.4.3 Kalman filter

Kalman Filtering is a recursive estimation method that is used to estimate the state of a dynamic system from a series of noisy measurements. It is widely utilized in control systems, robotics, signal processing, and sensor fusion.

In its core, the Kalman Filter predicts the current state of a system from a mathematical model and then refines this prediction from the available sensor data, taking into account the model and measurement uncertainties.

To enhance the reliability of predictions within the digital twin framework, a Kalman filter model was developed and implemented using Python. The Kalman filter is known for its effectiveness in estimating time-series data in dynamic systems, especially when measurements are noisy or incomplete.

A dedicated Kalman filter script was created to predict four key parameters of the neonatal incubator environment:

- Incubator Temperature
- Humidity
- Oxygen Concentration
- Infant Comfort Level

Each parameter was processed through a corresponding filter instance, allowing for continuous and real-time estimation based on historical and live sensor inputs. This approach helped in smoothing fluctuations and improving the accuracy of readings used in decision-making.

Figure IV.33 of the Kalman filter Python scripts for incubator environment prediction.

```python
# Prepare Kalman Filter
kf = KalmanFilter(dim_x=2, dim_z=1)
kf.x = np.array([0., 0.])  # initial state
kf.F = np.array([[1., 1.],
                 [0., 1.]])
kf.H = np.array([[1., 0.]])
kf.P *= 1000.  # initial uncertainty
kf.R = 0.01    # measurement noise
kf.Q = np.array([[1e-5, 0],
                 [0, 1e-5]])  # process noise
```

(1)

```python
kf = KalmanFilter(dim_x=2, dim_z=1)
kf.x = np.array([[oxygen[0]], [0]])
kf.F = np.array([[1, 1], [0, 1]])
kf.H = np.array([[1, 0]])
kf.P *= 1000
kf.R = 4
kf.Q = Q_discrete_white_noise(dim=2, dt=1, var=0.01)
```

(2)

```python
# Kalman Filter setup
n_features = X.shape[1]
kf = KalmanFilter(dim_x=2 + n_features, dim_z=1)
kf.x = np.zeros((2 + n_features, 1))
kf.F = np.eye(2 + n_features)
kf.H = np.zeros((1, 2 + n_features))
kf.H[0, 0] = 1
kf.R *= 0.01
kf.P *= 1.0
kf.Q = np.eye(2 + n_features) * 1e-4
```

(3)

```python
# Kalman Filter setup
n_features = X.shape[1]
kf = KalmanFilter(dim_x=2 + n_features, dim_z=1)
kf.x = np.zeros((2 + n_features, 1))  # initial state
kf.F = np.eye(2 + n_features)
kf.H = np.zeros((1, 2 + n_features))
kf.H[0, 0] = 1  # measure only first variable
```

(4)

Figure IV. 33: Kalman filter scripts, (1) temperature, (2) oxygen, (3) humidity, (4) baby-response.

1. Temperature prediction

Kalman Filter effectively captured the gradual incubator temperature changes, producing a predictive curve that followed the true measurements very closely with little lag. Scatterplot showed tight correlation, and the output of classification yielded high accuracy.

The classification output also achieved high accuracy. The results obtained are as follows: RMSE = 0.071, R^2 = 0.999, and Accuracy = 0.995 (see Figure IV.34).

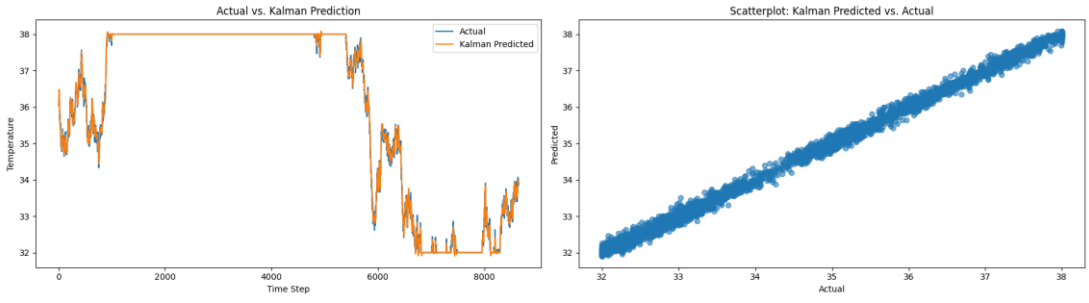


Figure IV. 34: Kalman Filter Temperature.

2. Oxygen concentration prediction

Oxygen level prediction through Kalman filtering was good with moderate error and considerable drop in the R^2 score, which measures lower goodness-of-fit. Accuracy, though, remained flawless due to very little variability in target labels. The classification output also

achieved high accuracy. The results obtained are as follows: figure IV.35 shows RMSE= 1.624 R^2 = 0.331Accuracy = 1.000.

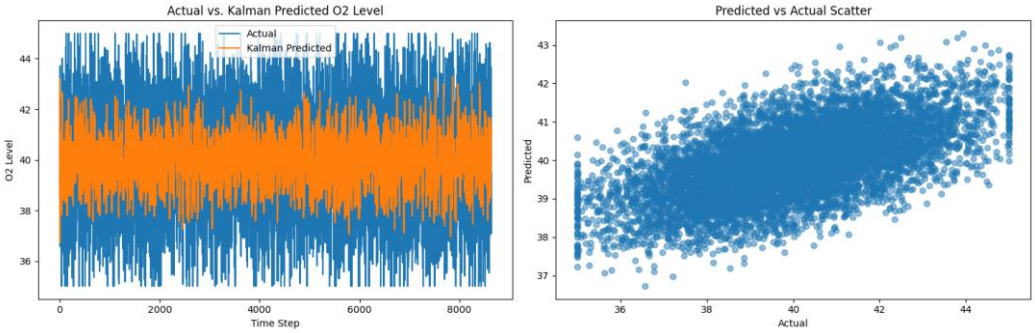


Figure IV. 35: Kalman filter oxygen.

3. Humidity prediction

Kalman filtering of humidity data resulted in increased RMSE and decreased R^2, suggesting challenges in adapting to rapid or noisy shifts in sensor values. However, the classification task still achieved full accuracy due to label homogeneity. The classification output also achieved high accuracy. The results obtained are as follows: figure IV.36 RMSE = 5.443 R^2 = 0.188 Accuracy = 1.000.

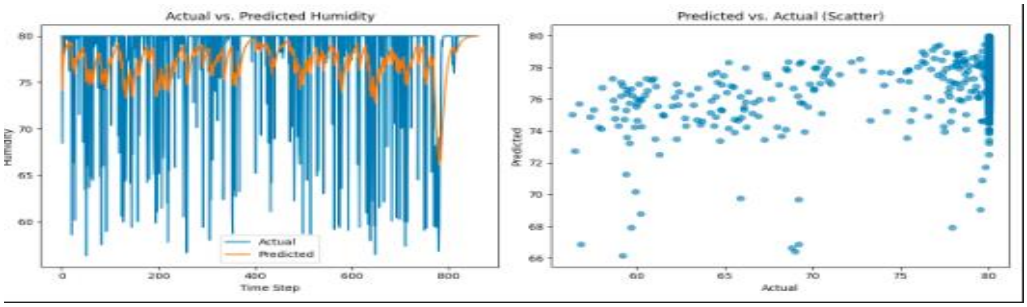


Figure IV. 36: Kalman filter humidity.

4. Baby response prediction

The estimation of infant response with RMSE showed comparatively low values but again, the R^2 score was limited, reflecting the Kalman Filter's cautious adjustment in rapidly changing behavioral signals. Similar to other parameters, classification accuracy was superb. The classification output also achieved high accuracy. The results obtained are as follows: figure IV.37, RMSE = 0.111 R^2 = 0.133 Accuracy = 1.000.

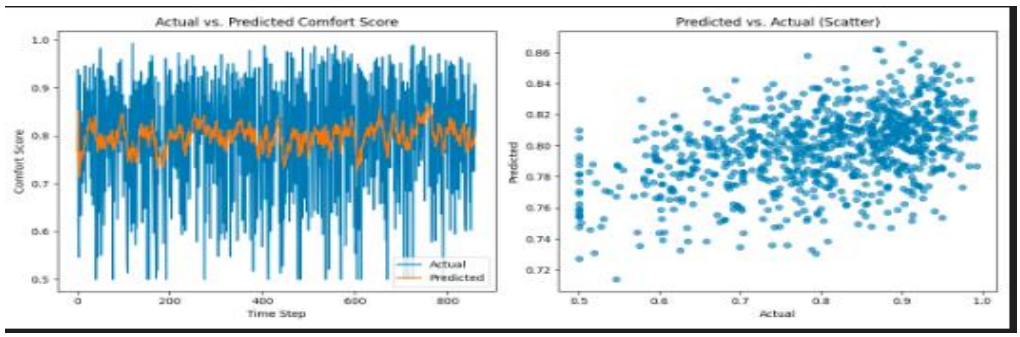


Figure IV. 37: Kalman filter baby response.

While the Kalman Filter worked extremely well in temperature prediction, with low RMSE and high R^2, it showed evidence of underfitting for humidity, oxygen, and baby response particularly in dynamic conditions. However, it still had perfect classification accuracy on all but the temperature variable, which was also high (0.995).

This would mean that the Kalman Filter is very good at dealing with smoothly varying systems (e.g., temperature), but would require some tweaking (e.g., Extended or Unscented Kalman variants) to handle non-linear or sudden behavioral signals like infant reaction or outside disruptions better

IV.4.4.4 Comparison of LSTM and Kalman

Filter results table IV:5 shows how both LSTM and Kalman Filter models were trained on four most critical variables in the neonatal incubator system: temperature, humidity, oxygen concentration, and baby response. The following is a general overview of results and analysis.

From the comprehensive performance measurements, the Kalman Filter outperformed the Bidirectional LSTM in all four prediction measures: temperature, oxygen level, humidity, and baby reaction. In the temperature prediction, the Kalman Filter registered a lower RMSE (0.071 vs 0.088157) and a higher R^2 score (0.999 vs 0.981414), indicating a precise and consistent estimation, even if the LSTM was able to achieve a perfect classification accuracy. In oxygen forecasting, the Kalman Filter again outperformed with lower RMSE (1.624 compared to 2.100185) and much better R^2 rating (0.331 compared to –0.015666) with equal classification precision (1.0). Forecasting in humidity established the largest difference: Kalman exhibited better regression metrics (RMSE of 5.443 compared to 6.036652, R^2 of 0.188 compared to 0.000651) and considerably better classification accuracy (1.0 compared to 0.857309). Finally, for the baby response variable, Kalman had a lower RMSE (0.111 vs 0.119955) and a greater R^2 value (0.133 vs –0.012450), both models achieving perfect accuracy. In general, the results confirm that while LSTM models were performing adequately, especially in tracking non-linear trends, the Kalman Filter was more stable and accurate particularly when handling smoothly varying signals so was the most reliable choice in this case.

We decided to use LSTMs in our project instead of relying solely on the Kalman filter due to several reasons. First, LSTM networks possess memory capabilities, allowing them to capture long-term dependencies and complex nonlinear temporal relationships. Second, our dataset was primarily composed of synthetic and expected values data we designed ourselves with limited variability and smooth transitions. In this controlled context, the Kalman filter actually outperformed LSTM by delivering better predictive accuracy, given its strength in modeling stable, linear systems. However, when dealing with real-world or noisy data such as fluctuating humidity, oxygen levels, or infant behavioral responses, LSTMs demonstrated superior adaptability. They learn patterns directly from the data without the need for predefined assumptions about system dynamics. Moreover, recent studies have shown that LSTM models tend to outperform Kalman filters in capturing irregularities and nonlinearities in physiological and environmental data. This made LSTM a more robust choice for modeling the unstable and sensitive environment inside a neonatal incubator.

Table IV-5: LSTM vs. kalman filter performance across key incubator variables

Variable	Model	RMSE	R^2 Score	Accuracy
Temperature	**LSTM**	0.088157	0.981414	1.0
	Kalman Filter	**0.0710**	**0.9990**	1.0
Oxygen	**LSTM**	2.100185	–0.015666	1.0
	Kalman Filter	**1.6240**	**0.3310**	1.0
Humidity	**LSTM**	6.036652	0.000651	0.857309
	Kalman Filter	**5.4430**	**0.1880**	1.0
Baby Resp	**LSTM**	0.119955	–0.012450	1.0
	Kalman Filter	**0.1110**	**0.1330**	1.0

IV.4.5 Web application development

The web application is built with React in a modular pattern. Normally located at the top or side of the screen is the navigation bar. The user can switch between the core modules of the application:

This provides the user flexibility to navigate between sections and always know their location and position in the application.

The web application consists of modular components, as shown in figure IV.39, each addressing a feature of the neonatal incubator monitoring and control system. The basic components are:

App.js: The root component that combines and displays all subcomponents and is the app's main entry point.

Main.js: The central layout controller that organizes the entire navigation and structure.

Navigation.js and BackButton.js: Provide smooth user navigation between app modules, including a consistent way to go back to previous screens.

Monitoring.js: Displays live incubator data retrieved from Firebase, such as temperature, humidity, and oxygen levels.

History.js: Loads and presents historical data from Firebase in table or graph format for backward analysis.

ControlPanel.js: Allows users to drive actuators (e.g., heating or ventilation systems on/off) for manual or automatic control.

Simulation.js: Provides means to simulate incubator environmental conditions and input different environmental conditions to test, for predictive modeling and system validation.

Settings.js: Serves user preferences such as theme (dark/light theme), language, notification options, and Firebase options.

View3D.js: Renders a 3D digital twin of the incubator from a `. glb` model imported using Three.js, giving an interactive view of the system's current state.

Drawings.js: Displays technical drawings, system design, and explanation diagrams to facilitate understanding.

SafetyAlert.js: Monitors system limits and notifies users upon detection of critical values or emergency conditions.

Overlay.js: Provides visual overlays for enhancing 3D interaction or highlighting specific elements within the interface.

firebaseConfig.js: Establishes Firebase connection, i.e., authentication and real-time database operations.

seedData.js: Initializes or loads test, simulation, or demonstration data sets.

Once development was complete, the app was created and executed locally with:

- cd my-app.
- npm start.

All these components work together in cohesion to create an interactive, intelligent, and responsive web platform for monitoring and controlling neonatal incubators.

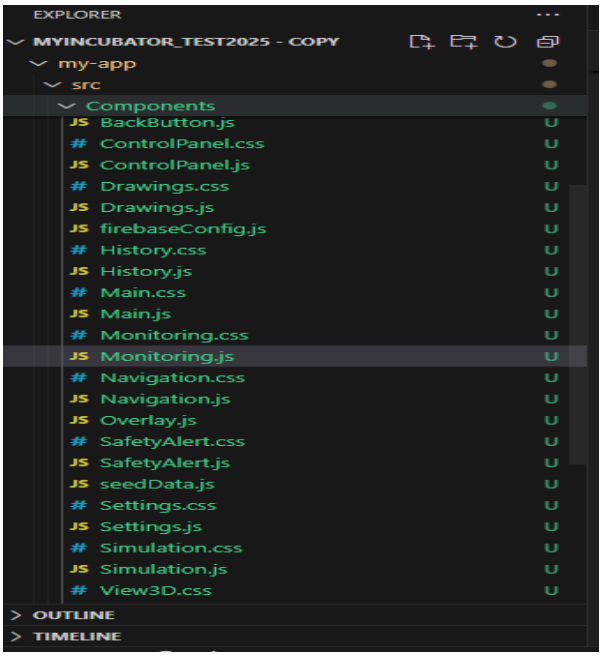


Figure IV. 38: Standard components.

IV.4.5.1 Main interface

The main interface (figure IV.40) is the main entry point to all critical functions of the web application. The interface is intended for effortless user experience aimed at providing simple, clear, and interactive experience that accommodates the requirements of monitoring, control,

simulation, and View3D of the neonatal incubator system. The main interface consists of the following core components:

Monitoring, history, control panel, simulation, view3D, settings, charts.

Normally located at the top or side of the screen is the navigation bar. The user can switch between the core modules of the application.

This provides the user flexibility to navigate between sections and always know their location and position in the application.

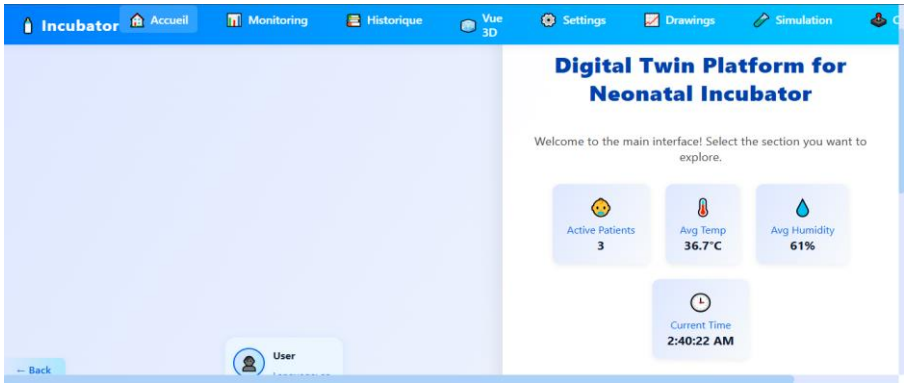


Figure IV. 39: Main interface.

IV.4.5.2 Monitoring Interface

The monitoring interface, illustrated in figure IV.41, serves as the primary dashboard for real-time observation of critical physiological and environmental parameters within the neonatal incubator. This interface plays a vital role in maintaining optimal conditions to ensure the health and comfort of the infant.

A. Key Functionalities

1. Real-Time Data Visualization

The interface continuously displays sensor readings in real time, covering the following parameters:

- Temperature.
- Humidity.
- Oxygen level.
- Heart rate.
- Weight.
- Respiratory rate.
- Blood pressure.
- Apnea episodes.
- Nutritional input.
- Behavioral indicators.

2. Firebase Integration

Live data is retrieved using Firebase functions such as onSnapshot () or getDocs (), depending on the configuration with Firestore or Realtime Database. These methods ensure that the displayed information is updated dynamically in response to changes in the sensor data.

3. Visual Status Indicators

The interface includes intuitive visual indicators to reflect the status of each parameter. For example, a change in indicator color (e.g., turning red) is used to signify values that deviate from the safe operating range, prompting immediate attention.

4. Safety Alert Mechanism

The interface is integrated with the SafetyAlert.js component, which is responsible for generating real-time warning messages when any monitored variable exceeds a predefined safety threshold. This mechanism ensures timely interventions in potentially hazardous situations.

5. Responsive Design

The monitoring dashboard is designed to adapt seamlessly across various screen sizes, including desktop, tablet, and mobile devices, thereby ensuring accessibility and usability in diverse clinical settings.

Technologies Employed

- **React Hooks**: Utilized (useEffect, useState) to manage state and side effects related to live data streaming.
- **Firebase SDK**: Used to synchronize sensor data with the interface in real time.
- **Tailwind CSS / Custom CSS:** Applied for modern and responsive UI styling.
- **Reusable Components:** Modular elements such as status cards and alert badges were implemented for maintainability and scalability.

B. User Experience

The monitoring interface enables medical personnel to:

- Instantly verify whether the incubator is functioning within safe operational limits.
- Receive timely alerts in the event of abnormal conditions.
- Rely on a robust, intelligent, and responsive monitoring system for neonatal care.



Figure IV. 40: Monitoring interface.

IV.4.5.3 History interface

This interface has also been designed to show the past sensor readings in our neonatal incubation system, as depicted by figures IV.42. It gives users the ability to access past readings of sensor data, such as temperature, humidity, oxygen concentration, and other vital parameters. This aspect allows physicians to track environmental and physiological changes over time.

With this interface, one is able to study the evolution of incubator internal conditions over a time period chosen by them. They might be able to observe temperature evolution, humidity evolution, and other significant evolution that can affect newborn health. The historical data visualisation is an overall view of incubator functioning over long time scales, which can result in better clinical decision-making.

For instance, by tracking the trends over time, users are able to identify known patterns which could indicate that system calibration or intervention at an early stage is needed. In addition, by comparing real-time data with historical data, system reliability can be tested and ongoing safety and comfort of the infant ensured.

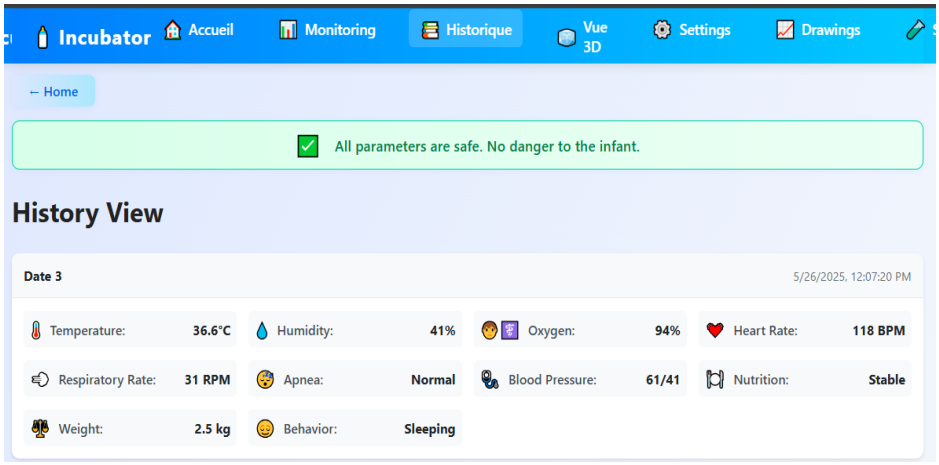


Figure IV. 41: History.

IV.4.5.4 Control interface

The Control Interface is a core component of our application, designed to give users instant control over the incubator's internal environment, as shown in the figure IV.43. Through this interface, users can manually enable or disable various triggers, such as the heating system, humidifier, oxygen supply, and lighting. This enables healthcare staff to make immediate adjustments based on the baby's current needs and the data displayed by the monitoring system.

The interface is user-friendly and clearly structured, allowing users to interact with the switches or buttons corresponding to each trigger. It also provides instant feedback on the status of each component, ensuring transparency and confidence in system operation. This layer of manual control complements the automated management logic, providing flexibility in emergency situations and precise control of incubator conditions.

By integrating real-time Firebase updates, any change made through the Control Interface is immediately reflected in the system database and across all other connected components. This ensures data consistency and seamless coordination between control actions and system responses.

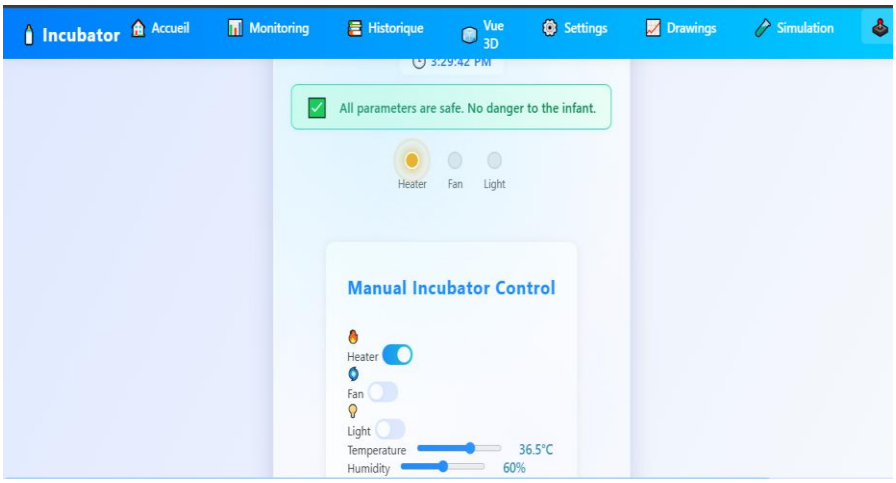


Figure IV. 42: Control interface.

IV.4.5.5 View3D interface

The View3D Interface is an interactive visualization component of our web application, providing a live 3D visual representation of the neonatal incubator setting. Aged from a changed 3D model originally developed in Blender and exported in `. glb` format, the interface enables users to navigate naturally through the incubator's parts, such as actuators and sensors, within a virtual context. With clickable hotspots, users can visualize sensor readings temperature, humidity, oxygen level, and baby activity projected onto their corresponding locations on the model (see figure IV.44).

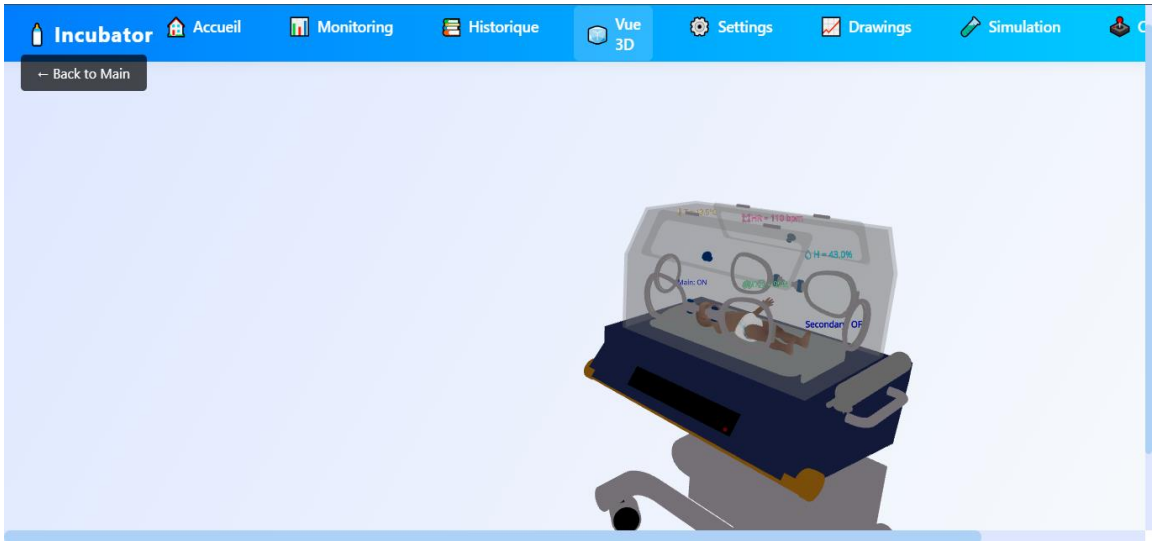


Figure IV. 43: View3D Interface.

This spatial interface more easily enables user interaction by showing the information with a sense of location, enabling a better understanding of the physical relationships among different parts of the system. It also enables dynamic updates: when the incubator is updated with new information from Firebase, the 3D view reflects these changes in near real time, simulating a real-time digital twin of the incubator.

By integrating technical accuracy and visual simplicity, View3D Interface bridges the chasm between data observation and control of the environment, and is especially beneficial for medical professionals to observe and analyze the incubator condition naturally and efficiently.

IV.4.5.6 Simulation interface

Simulation Interface is a control system that allows simulation of various operating modes of the neonatal incubator system. As can be seen in figure IV.44 and IV.44, this interface offers manual control of various actuators, e.g., heater, humidifier, fan, and oxygen valve, through interactive buttons. It finds particular application in system testing under defined conditions, independent of actual time sensor input.

This feature is essential during development, as it supports debugging, validation of control logic, and performance tuning. It also provides users such as medical practitioners or researchers with the ability to test in a safe environment what the system would do in the case of emergencies or out-of-the-ordinary environmental conditions. All actions in simulation are traced in the Firebase real-time database, offering traceability and correlation with previous data.

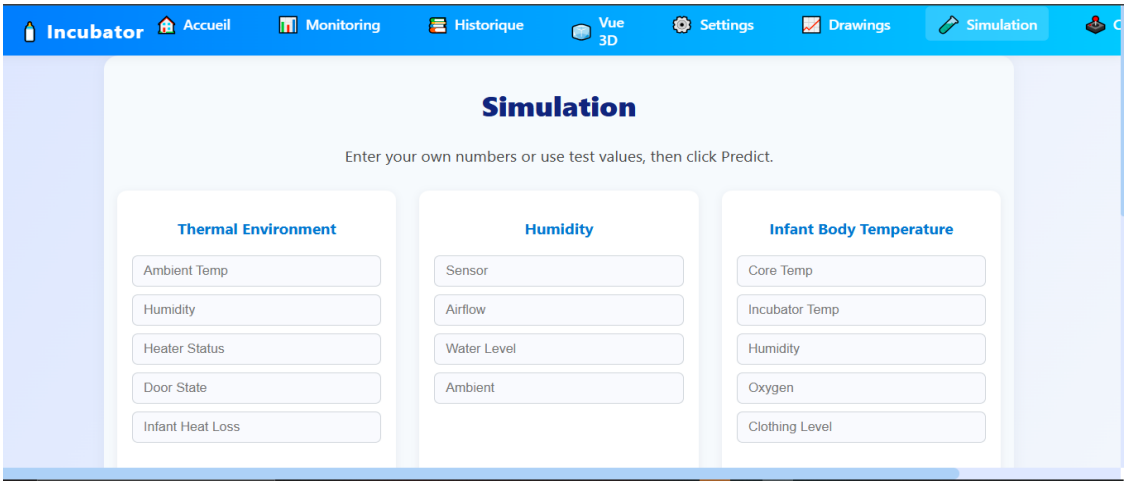


Figure IV. 44: Simulation interface (1).

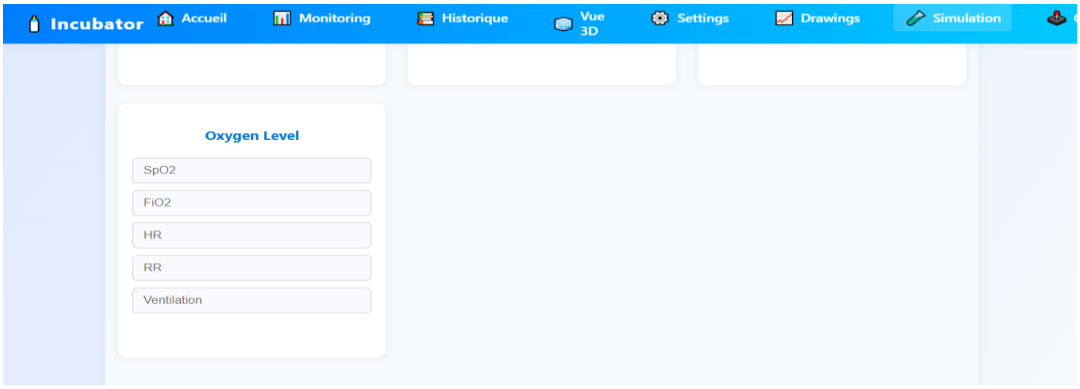


Figure IV. 45: Simulation interface (2).

Furthermore, when used in conjunction with the View3D Interface, the simulation is even more interactive and intuitive. Users can visualize the effect of their interactions in real-time mapped onto a 3D model, further establishing the concept of a digital twin and streamlining decision-making and user training.

IV.4.5.7 Settings interface

The Settings Interface is extremely valuable in personalizing and customizing the digital twin platform, based on user choices and operational standards. The interface provides options for changing various settings like system language, display mode (light/dark), notifications, and physical units of measurement for data related to sensors (like °C or °F%RH). These settings are stored in Firebase so the user preferences are saving and will be retained across sessions and devices.

The interface also allows for advanced settings to set operational thresholds for important parameters like temperature, humidity, and oxygen concentration. Setting the thresholds will flow through into the behavior of the monitoring and alert modules when changes are detected in the environment, leading to a real-time and/or automatic response. This affords a tailored system which is particularly responsive to the condition of each infant's context.

Because it is an easy to understand and navigate Settings Interface, it improves usability and responsivity, and is especially useful to medical personnel that need to have access quick and easy access to configuration tools as both a user and administrator in emergency situations.

An example of the Settings Interface is shown in figure IV.46.

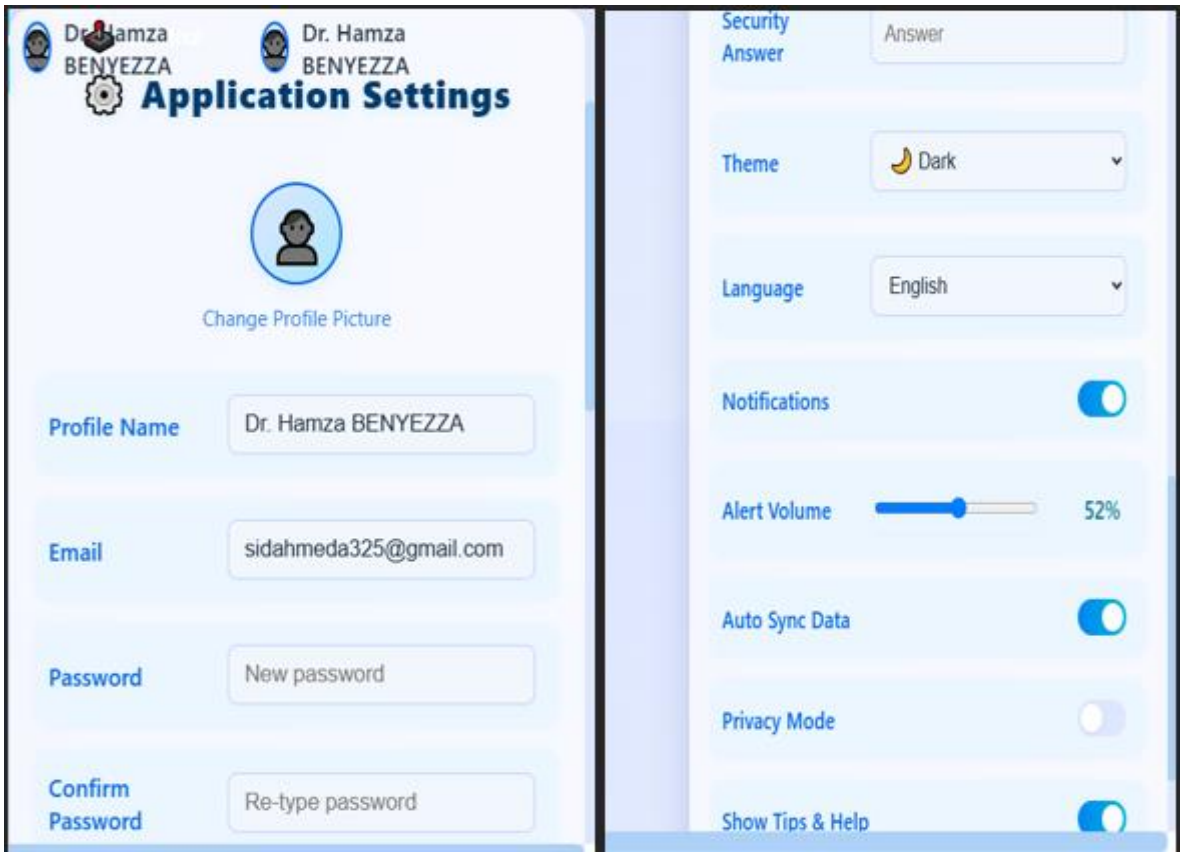


Figure IV. 46: Settings Interface.

IV.4.5.8 Charts interface

The Charts Interface, shown in figure IV.47, plays a crucial role within the application, offering the ability to visualize sensor data from the neonatal incubator in real-time, as well as historically. Examples of the key indicators presented include temperature, humidity, oxygen specs, and baby comfort measures, presented using familiar charts and graphs. The graphical visualization allows for improved understanding of the environmental trends available to clinicians so that they may react in a timely and informed manner.

The use of advanced charting libraries such as Recharts allows for interaction with the data, i.e., filtering by date, zooming into specific intervals, and comparing the evolution of variables over time. This activity contributes to predictive maintenance and health monitoring of the baby and solidifies a connection between purely sensor based outputs and meaningful medical information.

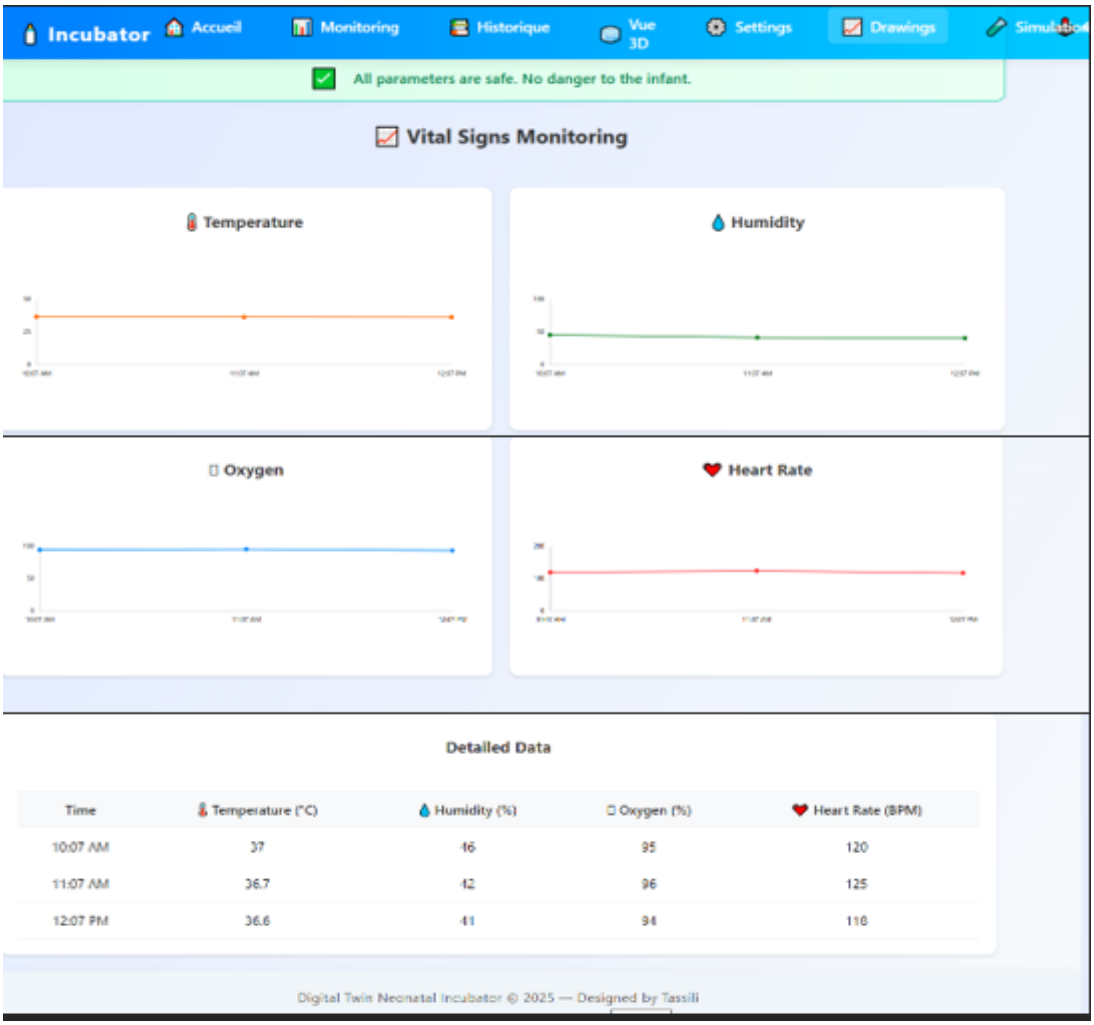


Figure IV. 47: Drawings Interface.

IV.5 Test results

To validate the suggested neonatal incubator management system, an actual incubator was not used due to its logistical and ethical constraints. A simulation-based test approach was employed in its place. Some of the incubator parameters such as temperature, humidity, oxygen level, and infant comfort indicators were simulated and fed into the system. The main intent of

this test was to validate the accuracy of data collection, real-time visualization, cloud synchronization, and predictive modeling.

IV.5.1 Validating data transfer and cloud storage

To test the data pipeline, we simulated sensor readings and streamed them to the cloud using an ESP32 microcontroller. The test's purpose was to confirm if the data was correctly captured and transmitted to Firebase Realtime Database in real time. figure IV.48 and IV.49, below demonstrate the live sensor data as it appears in the Realtime Database, confirming successful data synchronization from the device to the cloud backend.

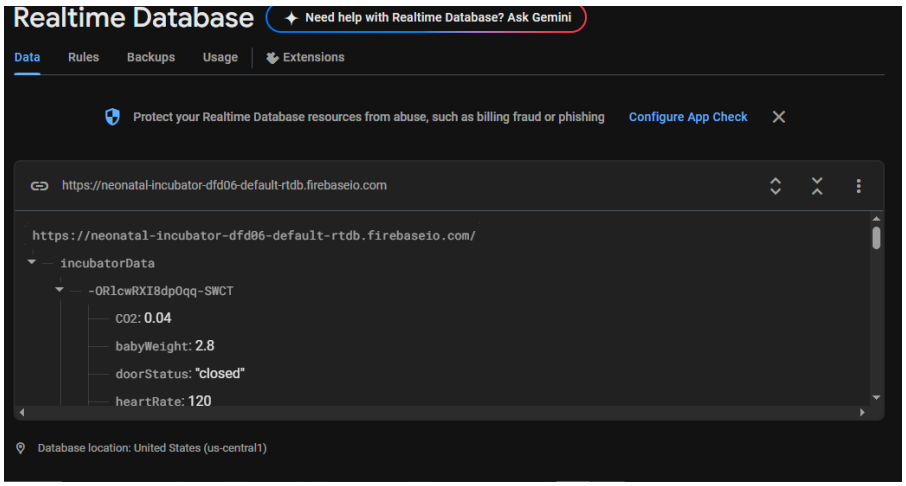


Figure IV. 48: Information stored in the Realtime Database (1).

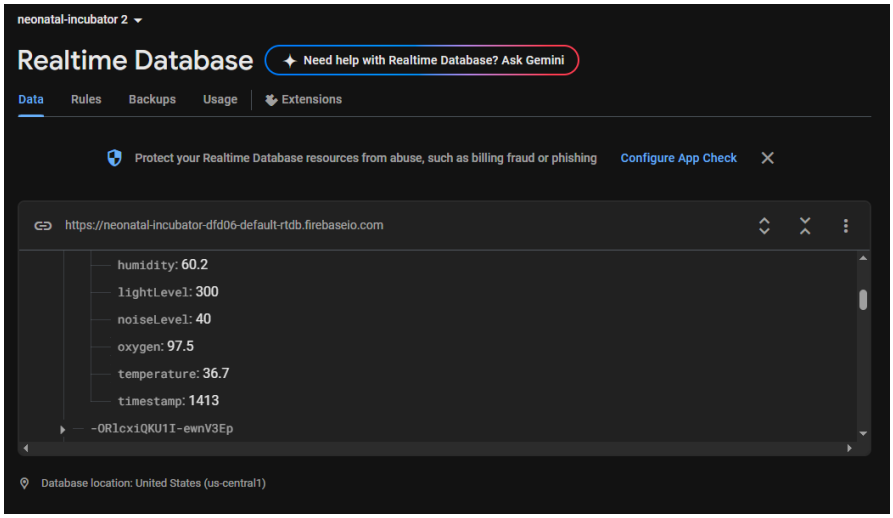


Figure IV. 49: Information stored in the Realtime Database (2).

IV.5.2 Simulated data entry and Firestore storage validation

In this phase, we manually entered simulated incubator data through the React web interface to validate the connection and data flow between the front-end and Cloud Firestore. As shown in figure IV.50 to IV.52, each document stored in the `incubatorData` collection contains a complete set of neonatal parameters, including apnea status, behavior, blood pressure, heart rate, humidity, nutrition status, oxygen saturation, respiratory rate, temperature, timestamp, and

weight. This step confirmed that our application successfully writes structured and organized data to Firestore, serving as a foundational validation of our data entry mechanism and cloud integration.

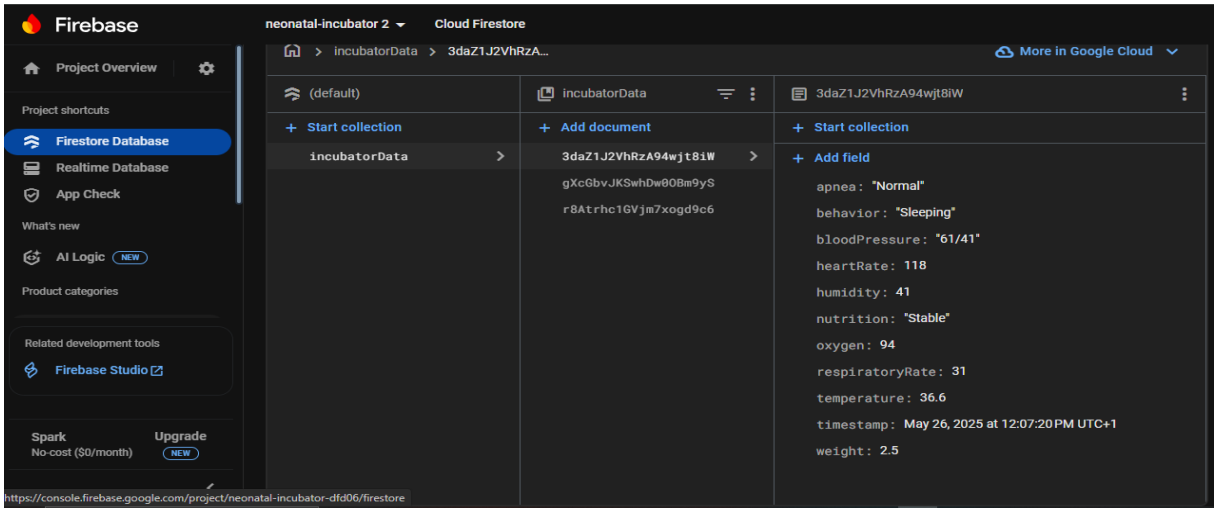


Figure IV. 50: Information stored in the Firestore (1).

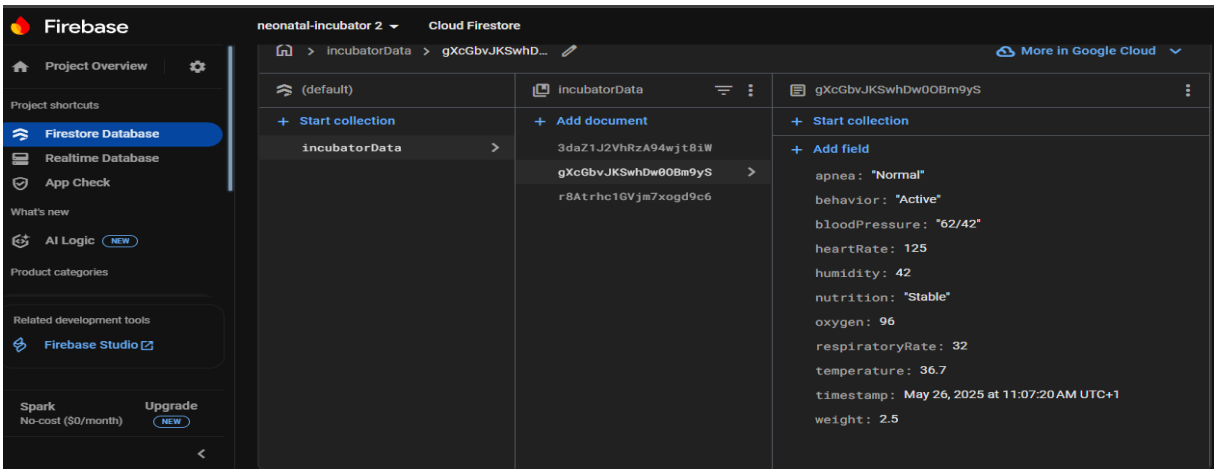


Figure IV. 51: Information stored in the Firestore (2).

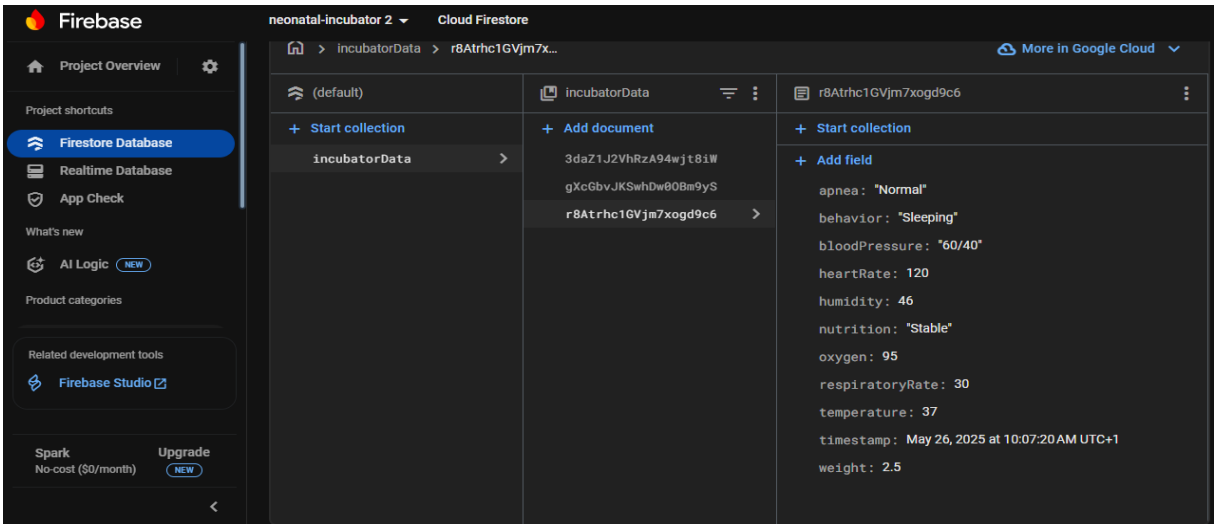


Figure IV. 52: Information stored in the Firestore (3).

IV.5.3 Simulation test

To enable interactive testing and performance analysis of predictive models, we developed a custom simulation interface using React. Using this interface, users can manually enter environmental and biometric parameters, such as ambient temperature, humidity, heater status, oxygen saturation, heart rate, and more. The interface is divided into four main sections: thermal environment, humidity, infant body temperature, and oxygen level. As illustrated in figure IV.53, each section collects relevant data and publishes it via HTTP POST requests to a local Flask server hosting the trained machine learning models. The models respond with predictions that are updated instantly on the interface. A "Fill test values" button is also available to populate fields with predefined values for easier testing. Basic validation is also performed to prevent the submission of abnormal or dangerous values (such as unrealistic heart rate or body temperature). This simulation tool is effective for evaluating the behavior of a digital twin system during operation and enables scenario-based testing without relying on physical hardware.

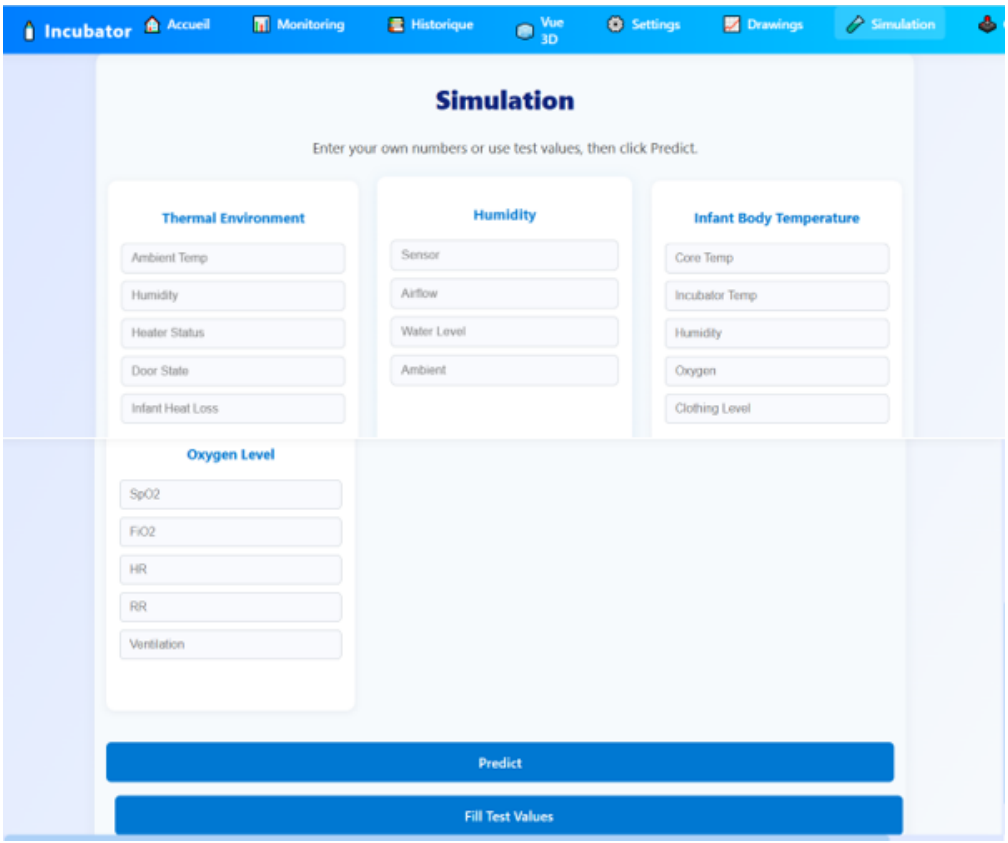


Figure IV. 53: Predictive results.

IV.5.4 Visualization 3D incubator

To allow for more user interaction and further the realism of output, a 3D model of the incubator was implemented into the system through the glb format. This test aimed at verifying that the 3D model accurately depicts sensor values in real time and offers a digital twin environment. The figure IV.54 shows the virtual incubator and how it displays dynamically changing readings, presenting users with a spatial and interactive representation of the incubator status.



Figure IV. 54: virtual incubator.

IV.6 Conclusion

In this chapter, we presented the whole design, development, and testing of an intelligent neonatal incubator management system according to the digital twin methodology. By combining Internet of Things (IoT) technologies, real-time data computing, 3D visualization, and predictive modeling using LSTM networks and Kalman filters, we created an interactive and responsive platform mimicking the physical incubator environment. The system architecture included layered sub-systems for data acquisition (using ESP32 and sensors), safe cloud storage (Firebase), dynamic prediction modules (LSTM/Kalman), and a modular React web interface augmented with real-time dashboards, simulation modules, and a 3D model of an incubator.

Our experiments demonstrated Bidirectional LSTM models to be most effective at predicting temperatures, while Kalman filters were improved more generally in regression precision across parameters with more stable conditions. Both modeling methodologies had weaknesses, though, in predicting dynamic variables such as humidity and infant behavioral response. This is in support of the utilization of hybrid models and other varied input parameters as means by which further improvement can be made.

The successful simulation-based testing verified the operability, scalability, and usability of the platform, promising a leap towards safer and more intelligent neonatal care. Overall, this digital twin approach gives a sound foundation to push neonatal monitoring forward with real-time decision support, predictive analytics, and remote control, in a user-centric and clinically relevant interface.

General Conclusion

General Conclusion

In this paper, we have successfully designed and implemented a DT platform for the control of neonatal incubators with embedded systems, real-time data acquisition, predictive modeling, and immersive visualization. The proposed architecture includes an ESP32 microcontroller for multi-sensor data collection, a Firebase cloud backend for real-time synchronization, and LSTM neural networks for predicting ambient conditions, especially temperature and humidity. Such predictions enable pre-emptive adjustment of incubator parameters to maintain optimal neonatal stability. Furthermore, a web-based 3D visualization interface developed with React and Blender enables real-time, dynamic interaction with the virtual twin, enhancing situational awareness among caregivers.

Results obtained demonstrate the platform's capacity to consistently predict temperature fluctuations and graph live sensor readings with an interactive and intuitive interface. The system was robust in collecting and transmitting live information and executing predictive simulations at low latency. However, the research also encountered some limitations, such as limited access to true neonatal data sets, and the need for improved real-time feedback mechanisms to counteract sudden physiological changes.

In the future, some adjustments could be included to further augment the system's capabilities. Firstly, the inclusion of real-time heart rate and pulse oximetry monitoring would provide valuable information about the baby's physiological state to identify stress or apnea episodes. Secondly, adaptive lighting control based on circadian rhythms or infant activity could further enhance neurodevelopment. Third, a smart alert and notification system based on anomaly detection algorithm could be incorporated to automatically alert healthcare personnel and parents via mobile or web portals whenever values exceed safe limits. It might also be augmented with sophisticated models such as bidirectional LSTM, attention, or even hybrid PINNs to enhance the precision and interpretability of predictions. Finally, the site could also be extended to include remote control functionality, for example, actuator command through the web interface to manage temperature, light, humidity, and oxygen in real time.

In brief, this project offers the basis for the future intelligent neonatal incubator system based on digital twin technology to offer intelligent, responsive, and personalized neonatal care. It is highly promising for future clinical use and research, in line with the Healthcare 4.0 vision and overall movement towards predictive, preventive, and patient-enabled medicine.

References

References

[1] World Health Organization. *Preterm birth.* WHO [online]. [Accessed 4 May 2025]. Available at: https://www.who.int/fr/news-room/fact-sheets/detail/preterm-birth

[2] World Health Organization. *Kangaroo mother care: a practical guide.* WHO [online]. 2003. [Accessed 4 May 2025]. Available at: https://europepmc.org/article/med/23420913

[3] TAO, Fei, QI, Q., LIU, A., & KUSIAK, A. Data-driven smart manufacturing. *Journal of Manufacturing Systems*, 2018, vol. 48, p. 157–169.

[4] BIBBY, Laura & DEHE, Benjamin. Healthcare 4.0. *International Journal of Operations & Production Management*, 2018, vol. 38, n°1, p. 66–85.

[5] LAWN, Joy E., et al. Every Newborn: progress, priorities, and potential beyond survival. *The Lancet*, 2014, vol. 384, n°9938, p. 189–205.

[6] Lester, Barry M., Hawes, K., Abar, B., Sullivan, M., Miller, R., Bigsby, R., et al. *The Neonatal Intensive Care Unit: Environmental Stressors and Supports.* Clinical Perinatology [online]. 2018, Vol. 45, No. 3, pp. 539–555. [Accessed 4 May 2025]. Available at: https://pmc.ncbi.nlm.nih.gov/articles/PMC5800159/

[7] Peng, Nai-Hui, Bachman, J., Jenkins, R., Chen, C.-H., Chang, Y.-C., Chang, Y.-S., Wang, T.-M. *Relationships between environmental stressors and stress biobehavioral responses of preterm infants in NICU.* Advances in Neonatal Care [online]. 2009, Vol. 23, No. 4, pp. 363–371. [Accessed 4 May 2025]. Available at: https://pubmed.ncbi.nlm.nih.gov/19915421/

[8] Wuni, Abubakari, Iddrisu, O. A., Chanayireh, L., et al. *Environmental stressors, coping mechanisms and support system for mothers with infants at the neonatal intensive care unit: a descriptive cross-sectional study.* PAMJ-One Health [online]. 2022, Vol. 9, Article 12. [Accessed 4 May 2025]. Available at: https://www.one-health.panafrican-med-journal.com/content/article/9/12/full/

[9] Dräger Medical. *The Caleo® Effect – Innovation in Thermoregulation.* [online]. [Accessed 4 May 2025]. Available at: https://www.draeger.com/en_me/Products/Caleo

[10] ProCare Medical Furniture. *ProCare Servo Controlled Infant Incubator.* [online]. 2024. [Accessed 4 May 2025]. Available at: https://procare.com.pk/product/procare-infant-incubator/

[11] Dräger. Isolate® 8000 Neonatal Closed Care. [online]. 2025, [accessed on May 4, 2025]. Available at: https://www.setunari.com/files/editor/source/isolette-8000-pi-9069066-en-master-1609-3.pdf.

[12] E. H. Glaessgen and D. S. Stargel, “The digital twin paradigm for future NASA and U.S. Air force vehicles,” Collect. Tech. Pap. - AIAA/ASME/ASCE/AHS/ASC Struct. Struct. Dyn. Mater. Conf., no. April, 2012, doi: 10.2514/6.2012-1818.

[13] M. Grieves, “Digital Twin : Manufacturing Excellence through Virtual Factory Replication – A Whitepaper by Dr . Michael Grieves,” White Pap., no. March, pp. 1–7, 2014.

[14] P. Hehenberger and D. Bradley, Mechatronic Futures.

[15] L. Wang, M. Törngren, and M. Onori, "Current status and advancement of cyber-physical systems in manufacturing," J. Manuf. Syst., vol. 37, pp. 517–527, 2015, doi: 10.1016/j.jmsy.2015.04.008.

[16] "Grieves, M.W. Product lifecycle management: The new paradigm for enterprises. Int. J. Prod. Dev. 2005, 2, 71–84".

[17] E. A. Lee, "Computing Foundations and Practice for Cyber- Physical Systems : A Preliminary Report," 2007.

[18] "Grieves M. Virtually Perfect: Driving Innovative and Lean Products Through Product Lifecycle Management. Cocoa Beach, FL, USA: Space Coast Press; 2011".

[19] E. J. Tuegel, A. R. Ingraffea, T. G. Eason, and S. M. Spottswood, "Reengineering aircraft structural life prediction using a digital twin," Int. J. Aerosp. Eng., vol. 2011, 2011, doi: 10.1155/2011/154798.

[20] "Glaessgen EH, Stargel DS. The Digital Twin Paradigm for Future NASA and U.S. Air Force Vehicles. https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20120008178.pdf".

[21] J. Lee, B. Bagheri, and H. A. Kao, "A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems,"Manuf.Lett.,vol. 3, pp. 18–23, 2015, doi: 10.1016/j.mfglet.2014.12.001.

[22] "Grieves M. Digital twin: Manufacturing excellence through virtual factory replication. Digital Twin White Paper. 2014. http://innovate.fit.edu/plm/documents/doc_mgr/912/ 1411.0_Digital_Twin_White_Paper_Dr_Grieves.pdf".

[23] "Warwick G. GE advances analytical maintenance with digital twins. Aviation Week & Space Technology. 2015".

[24] M. Dr. Grieves, "Digital Twin: Manufacturing Excellence through Virtual Factory Replication," White Paper, March 2014. [Online]. Available: https://www.researchgate.net/publication/275211047_Digital_Twin_Manufacturing_Excellence_through_Virtual_Factory_Replication.

[25] https://netmoregroup.com/digital-twin-explained/[Accessed:13-Mai-2023].

[26] "The digital twin: plotting the course to automation." [Online]. Available: https://www.seatrade- maritime.com/technology/digital-twin-plotting-course-automation

[27] "The Digital Twin Drives Smart Manufacturing." [Online]. Available: https://www.arcweb.com/industry- best-practices/digital-twin-drives-smart-manufacturing

[28] "Digital cities-digital twins for connected city infrastructure." [Online]. Available:

https://www.geospatialworld.net/blogs/digital-cities-digital-twins-for-connected-city-infrastructure/

[29] "Digital Twins in Aviation and 8 Ways They Can Transform Workflows." [Online]. Available:https://kontakt.io/blog/digital-twins-in-aviation-and-8-ways-they-can-transform-workflows/

[30] T. Sun, X. He, and Z. Li, “Digital twin in healthcare: Recent updates and challenges,” Digit. Heal., vol. 9, 2023, doi: 10.1177/20552076221149651

[31] Stankovic, J.A. (2014). Research directions for the internet of things. IEEE Internet of Things Journal, 1(1): 3-9.

[32] Amazon Web Services (AWS). *"AWS IoT TwinMaker Documentation."* [Accessed 13May 2025] Available at: https://docs.aws.amazon.com/iot-twinmaker.

[33] Microsoft Azure. *"Azure Digital Twins Documentation." [Accessed 13 May 2025]. Availableat:* https://learn.microsoft.com/en-us/azure/digital-twins/

[34] Eclipse Foundation. *"Eclipse Ditto – Digital Twin framework for IoT." [Accessed 13 May 2025].Availableat:*https://www.eclipse.org/ditto/

[35] Gathid. (2024). *Digital Twins: The New Frontier in Cybersecurity*. Retrieved from [Accessed 13 May 2025]. Available at:https://gathid.com/news/digital-twins-the-new-frontier-in-cybersecurity/

[36] Python Software Foundation, "The Python tutorial," Version 3.13.3, 2025. [Online]. Available: https://docs.python.org/3/tutorial/index.html. [Accessed: May 20, 2025].

[37]Blender Foundation. (2025). *Features*. Blender.org. Retrieved May 21, 2025, from https://www.blender.org/features/

[38] SAXENA, Archit. *Introduction to Long Short-Term Memory (LSTM)*. Medium – Analytics Vidhya, 2020. [En ligne]. Disponible sur : https://medium.com/analytics-vidhya/introduction-to-long-short-term-memory-lstm-a8052cd0d4cd (consulté le 14 mai 2025).

[39] GeeksforGeeks. *Understanding of LSTM Networks*. GeeksforGeeks, 5 juin 2023. [En ligne]. Disponible sur : https://www.geeksforgeeks.org/understanding-of-lstm-networks/ (consulté le 14 mai 2025).