A Minor Project Report

on

IOT & ML BASED HEALTH MONITORING WEB APP

In the partial fulfilment for the award of the degree of

Bachelor of Technology

in

COMPUTER SCIENCE AND ENGINEERING

Submitted by

Shaik A.N.A.La Ilahi	**RO200249**
K. Veni Geethika	**RO200101**
K. Treshika Shailu	**RO201081**
M. Siva Durga	**RO200966**

Under the guidance of

Mr.T.SREEKANTH M.Tech.,(PhD)

Assistant Professor

Computer Science and Engineering

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

RAJIV GANDHI UNIVERSITY OF KNOWLEDGE TECHNOLOGIES ONGOLE

2024-2025

RAJIV GANDHI UNIVERSITY OF KNOWLEDGE TECHNOLOGIES

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that the project report entitled "IoT & ML Based Health Monitoring Web App" Submitted by Shaik Abdul Nabiul Alam La Ilahi (RO200249), K. Veni Geethika (RO200101), K. Treshika Shailu (RO201081), M. Siva Durga (RO200966) partial fulfilment of the requirement for the award of Bachelor of Technology in Computer Science and Engineering is a record of bonafide project work carried out under my supervision during the academic year 2024-2025.

We are indebted to Mr.T. Sreekanth M.Tech., (PhD), Our project guide for conscientious guidance and encouragement to accomplish this project. I extremely thankful and pay my gratitude to Mr.N.Mallikarjuna M.Tech(I/C) Head of the Department CSE, for this valuable guidance and support on the completion of this project.

The report hasn't been submitted previously in part or full to this or any other university or institution for the award of any degree.

Mr.T. Sreekanth M.Tech., (Ph.D),
Assistant Professor,
Department of CSE,
RGUKT, ONGOLE.

Mr.N. Mallikarjuna M.E.,(Ph.D)
Assistant Professor,
Head of the Department(I/C),
Department of CSE,
RGUKT, ONGOLE

RAJIV GANDHI UNIVERSITY OF KNOWLEDGE TECHNOLOGIES

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project report entitled "IoT & ML Based Health Monitoring Web App" submitted by Shaik A.N.A.La Ilahi (RO200249), K. Veni Geethika (RO200101), K. Treshika Shailu (RO201081), M. Siva Durga (RO200966) to the Department of Computer Science and Engineering, Rajiv Gandhi University of Knowledge Technologies, Ongole, during the academic year 2024-2025 is a partial fulfilment for the award of Under graduate degree of Bachelor of Technology in Computer Science and Engineering, is a bonafide work record carried out by them under my supervision during the academic year 2024-25.

The Report hasn't been submitted previously in part or in full to this or any other university or institution for the award of any degree.

Mr.T. Sreekanth M.Tech., (Ph.D),
Assistant Professor,
Department of CSE,
RGUKT, ONGOLE.

Mr.N. Mallikarjuna M.E.,(Ph.D)
Assistant Professor,
Head of the Department(I/C),
Department of CSE,
RGUKT, ONGOLE

APPROVAL SHEET

This report entitled "IoT & ML Based Health Monitoring Web App" submitted by Shaik A.N.A.La Ilahi (RO200249), K. Veni Geethika (RO200101), K. Treshika Shailu (RO201081), M. Siva Durga (RO200966) to Mr.T. Sreekanth M.Tech., (Ph.D), Assistant Professor, RGUKT, Ongole approved for the degree of Bachelor of Technology in COMPUTER SCIENCE AND ENGINEERING.

Examiner

____

____

Supervisor

____

____

____

____

Date: ______________________

Place: ______________________

DECLARATION

We declare that this written submission represents our ideas in our own words where other ideas or words have been included, We have adequately cited and referenced the original sources. We also declare that We have adhered to all principles of academic honestly and integrity and have not misrepresented or fabricated or falsified any idea /data/fact/source in my submission.

We understand that any violation of the above will be cause for disciplinary action by the institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Signature

SHAIK A.N.A.LA ILAHI (RO200249) ____________________
K. VENI GEETHIKA (RO200101) ____________________
K. TRESHIKA SHAILU (RO201081) ____________________
M. SIVA DURGA (RO200966) ____________________

DATE: ____________________
PLACE: ____________________

ACKNOWLEDGEMENT

We take this opportunity to express deep gratitude to the people who have been instrumental in the successful completion of the project.

We would like to thank our project guide **Mr.T.SREEKANTH M.Tech.,(PhD)** which was greatly helpful in the progression and smoothness of the entire project work. We would like to show our gratitude to our beloved Head of the Department, **Mr.N.Mallikarjuna M.E.,(PhD)** for his

encouragement in the aspect of our course, **RAJIV GANDHI UNIVERSITY OF KNOWLEDGE TECHNOLOGIES,** who gave us official support for the progress of our project.

We are deeply indebted to pay our sincere thanks to our Academic Dean **Mr. M. RUPAS KUMAR**, Asst. Professor , Dept. Of Civil Engineering,for showing some exuberant consent for this project and providing me with all the facilities in the department to complete the project.

We convey our profound thanks and gratitude to **Prof. BHASKAR PATEL**, Director of RGUKT ONGOLE for providing me an excellent academic climate which made this endeavour possible.

We would also like to extend our gratitude and sincere thanks to all the faculty members and Non-Teaching Staff in the Department of Computer Science and Engineering for supporting us.

We would also like to thank our parents and friends, who have willingly helped us out with their abilities in completing the project work.

With Sincere Regards

SHAIK A.N.A.LA ILAHI (RO200249)

K. VENI GEETHIKA (RO200101)

K. TRESHIKA SHAILU (RO201081)

M. SIVA DURGA (RO200966)

Date:________________________

ABSTRACT

This project introduces a real-time health monitoring and diagnostic system that synergizes IoT, machine learning, and generative AI to deliver intelligent, accessible healthcare solutions. At its core is the ESP32 microcontroller, connected to vital sensors like the MAX30100 (for heart rate and SpO_2) and a temperature sensor, enabling the continuous collection of health data. This data is broadcasted over a local network via an onboard web server. A lightweight Streamlit web application periodically fetches the sensor data, feeds it to a trained Random Forest Classifier, and determines the user's health status as Healthy or Unhealthy. The system further utilizes Google's Generative AI (Gemini API) to automatically generate a personalized health report with prescription-like insights, which can be downloaded as a text file. This integrated, AI-powered diagnostic platform enhances remote monitoring, reduces the dependence on manual checkups, and opens new possibilities for proactive, home- based, and rural healthcare solutions.

TABLE OF CONTENTS

Titles **Page No**

1. INTRODUCTION

Machine Learning

In today's world, technology is evolving at an incredible pace. One of the most fascinating and powerful advancements in this journey is Machine Learning (ML). Machine Learning is not just a buzzword; it is a revolutionary approach that allows computers to learn from experience, just like humans do. Rather than following strict rules programmed by humans, machines are now capable of understanding patterns, making decisions, and even improving themselves over time.

What is Machine Learning?

At its core, Machine Learning is a field of computer science and artificial intelligence that focuses on building systems that can learn from data. Instead of explicitly telling a computer what to do for every possible situation, we give it large amounts of data and allow it to figure out the best actions on its own.

In simple terms, Machine Learning is about teaching machines to recognize patterns and make informed decisions — much like how we humans learn from experience.

For example, think about how a child learns to recognize different animals. No one writes down a giant rulebook saying, "If it has four legs and fur, it's a dog." Instead, the child sees many examples and learns to recognize dogs over time. Similarly, in Machine Learning, we feed the machine many examples, and over time, it figures out how to classify or predict outcomes.

Why is Machine Learning Important?

Machine Learning has become important because of three main reasons:

- Data Explosion: Every day, we produce enormous amounts of data — from social media, healthcare records, financial transactions, and more. ML helps in making sense of this data.
- Computational Power: Advances in hardware (like powerful GPUs) allow machines to process and analyze vast amounts of information quickly.
- Improved Algorithms: Researchers have developed smarter techniques that allow computers to learn better, faster, and more accurately.

Thanks to Machine Learning, we now have self-driving cars, voice assistants like Siri and Alexa, personalized recommendations on Netflix and Amazon, advanced medical diagnostics, and even AI that can compose music or write poetry.

predict outcomes for new, unseen data.

Types of Machine Learning

Machine Learning is not a one-size-fits-all concept. It comes in different forms depending on how the learning happens:

1. Supervised Learning

This is the most common type. Here, the machine is trained on labeled data — meaning, the input and the correct output are both provided.

Example: Training a model to recognize cats and dogs by showing it thousands of pictures labeled "cat" or "dog."

The model learns the relationship between inputs and outputs and can then predict outcomes for new, unseen data.

2. Unsupervised Learning

In unsupervised learning, we do not provide any labels. The machine has to find patterns and structures in the data on its own.

Example: Grouping customers based on their buying behavior without knowing in advance what categories exist.

This is useful for clustering, anomaly detection, and finding hidden structures in data.

3. Reinforcement Learning

Here, the machine learns by interacting with its environment and receiving feedback in the form of rewards or punishments.

Example: Training a robot to walk by rewarding it when it moves correctly and penalizing it when it falls.

Over time, the robot learns the best strategy to achieve its goal.

How Does Machine Learning Work?

The process of Machine Learning typically follows these steps:

1.Data Collection:

Everything starts with gathering data. The quality and quantity of data are crucial because the model's learning is based on this data.

2.Data Preparation:

The raw data is cleaned and organized into a usable format. Missing values are handled, noise is reduced, and relevant features are selected.

3.Model Selection:

Depending on the problem (classification, regression, clustering, etc.), a suitable machine learning model is chosen.

4.Training the Model:

The model is trained on the prepared data, learning the patterns that link inputs to outputs.

5.Evaluation:

The model is tested on new, unseen data to see how well it performs. Metrics like accuracy, precision, recall, and F1-score are used for evaluation.

6.Deployment and Improvement:

Once the model performs well, it is deployed into the real world. However, learning never stops. Models are continuously monitored and improved over time as more data becomes available.

Real-Life Applications

- Healthcare: Predicting diseases early.
- Finance: Detecting fraud and improving investments.
- Entertainment: Personalized movie, music, and shopping recommendations.
- Transportation: Powering autonomous vehicles.

Internet of Things

The world around us is becoming smarter every day — not just because of new gadgets, but because these gadgets are starting to talk to each other. This incredible change is made possible by the Internet of Things (IoT). IoT is quietly reshaping how we live, work, travel, and even take care of our health. It's not a distant dream anymore; it's happening all around us, often without us even noticing.

What is the Internet of Things (IoT)?

At its core, the Internet of Things refers to a vast network of physical devices connected to the Internet, all collecting and sharing data.

These devices — ranging from smartwatches and refrigerators to industrial machines and medical sensors — can sense their environment, exchange information, and even act intelligently without human intervention.

In simple words, IoT is about connecting everyday objects to the Internet, allowing them to send and receive data, making our world more responsive and interactive.

For example, a smart thermostat that learns your schedule and adjusts your home's temperature automatically is an IoT device. So is a fitness tracker that monitors your heart rate and uploads the data to your phone.

Why is IoT Important?

IoT has gained such massive importance because of three major factors:

- Ubiquity of Internet Access: With Wi-Fi and mobile data available almost everywhere, devices can easily stay connected.

- Advancements in Sensors and Hardware: Modern sensors are smaller, cheaper, and more powerful than ever.
- Data-Driven Decision Making: Real-time data from IoT devices enables smarter, quicker decisions across industries.

Thanks to IoT, we now have smart homes, connected cars, automated factories, remote health monitoring, and even "smart cities" that manage traffic and utilities efficiently.

Key Components of IoT

IoT systems are made up of several important building blocks:

1. Devices/Sensors:
2. Physical objects that gather data from the environment (like temperature, heart rate, motion, etc.).
3. Connectivity:
4. Devices need to connect to the Internet, usually via Wi-Fi, Bluetooth, cellular networks, or newer technologies like LoRaWAN.
5. Data Processing:
6. After data is collected, it needs to be processed. Sometimes it's done on the device itself (edge computing) or sent to a cloud server for analysis.
7. User Interface:
8. Finally, users need a way to interact with the system, whether it’s a mobile app, a web dashboard, or automated alerts.

How Does IoT Work?

Here’s a simple breakdown of the typical IoT workflow:

- Sensing: Devices sense and collect data from their surroundings.
- Sending: This data is transmitted via the Internet.
- Processing: Data is analyzed, often using AI or ML to find patterns or anomalies.
- Action: Based on the analysis, actions are taken — either automatically or with human involvement.

For example, a smart irrigation system can monitor soil moisture and automatically water plants when needed, saving both water and energy.

Real-Life Applications of IoT

IoT is transforming countless areas of our lives:

- Smart Homes: Lights, security systems, and appliances that can be controlled remotely.
- Healthcare: Wearable devices that monitor vital signs and send alerts for abnormalities.

- Transportation: Connected vehicles that communicate with each other to avoid accidents.
- Agriculture: Sensors in fields that track soil health, humidity, and crop conditions.
- Industrial IoT (IIoT): Machines in factories that predict failures before they happen, improving efficiency.

1.1 About our Project

In today's fast-paced world, access to timely and accurate health information can make a huge difference — especially in areas with limited medical facilities.

Our project, "Real-Time Health Monitoring System using IoT, Machine Learning, and Generative AI," aims to bridge this gap by creating a lightweight, portable solution for continuous health tracking.

Using an ESP32 microcontroller connected to sensors like the MAX30100 and a temperature sensor, our system captures essential health parameters such as heart rate, blood oxygen levels (SpO_2), and body temperature.

This data is transmitted over a local Wi-Fi network through the ESP32's web server, ensuring real-time accessibility without depending on external servers.

A Streamlit-based web application retrieves this data, processes it using a trained Random Forest machine learning model, and predicts the user's health status.

To add a personalized touch, we integrated Google's Gemini API, which generates an AI-powered health report summarizing the user's condition. This report can also be easily downloaded as a PDF for record-keeping or remote consultations.

By combining IoT, Machine Learning, and Generative AI into a single platform, this system offers an innovative, user-friendly, and efficient approach to health monitoring — particularly useful for remote areas where access to healthcare might be limited.

1.2 Motivation of the Project

This project introduces a real-time health monitoring system that integrates IoT, Machine Learning, and Generative AI for proactive diagnostics. Using an ESP32 microcontroller with sensors like MAX30100 and a temperature sensor, it collects vital signs and serves them via a local web server. A Streamlit app fetches this data, predicts health status using an ML model, and generates an AI-based report via Google Gemini, which can be downloaded as a PDF for remote healthcare support.

1.3 Problem definition

Access to timely and affordable healthcare is limited, especially in remote areas. Traditional health monitoring is manual and infrequent, often leading to delayed diagnosis. There’s a lack of intelligent systems that can continuously track vital signs, analyze them in real-time, and provide easy-to-understand health insights. This project addresses the need for a low-cost, real-time health monitoring system using ESP32, machine learning, and generative AI to deliver instant health status predictions and AI-generated reports for better remote care.

1.4 Objectives of project

The primary goal of this project is to create a smart, real-time health monitoring system that empowers users with instant access to their vital health data and intelligent diagnostic feedback. By integrating IoT technology, machine learning, and generative AI, the system aims to enhance healthcare accessibility, especially for remote and underserved areas. The specific objectives include:

- To design a real-time health monitoring system using the ESP32 microcontroller and biomedical sensors for continuously tracking heart rate, SpO_2, and body temperature.
- To implement a machine learning model that classifies the user's health condition as “Healthy” or “Unhealthy” based on real-time sensor data.
- To develop an interactive and user-friendly Streamlit web interface that displays live vital signs and prediction outcomes.
- To integrate the Google Gemini API for generating AI-assisted, context-aware health reports and basic medical advice.
- To provide functionality for users to download the AI-generated report as a text file for documentation or consultation. To deliver a cost-effective, portable, and scalable solution suitable for home-based, rural, or remote health monitoring needs.

2. LITERATURE SURVEY

2.1. Literature Review

[A]. Nabila Sabrin Sworna, A.K.M. Muzahidul Islam, Swakkhar Shatabda, Salekul Islam, Towards development of IoT-ML driven healthcare systems: A survey, Journal of Network and Computer Applications,Volume 196, 2021, 103244, ISSN 1084-8045.

The impact of IoT-ML in the healthcare sector is very significant and it has helped us to change our view at the traditional treatment methods. In IoT-ML-based healthcare applications, the sensing layer is responsible for collecting information from humans and transferring it to the storage layer through communication technology.

[B]. H. K. Bharadwaj et al., "A Review on the Role of Machine Learning in Enabling IoT Based Healthcare Applications," in IEEE Access, vol. 9, pp. 38859-38890, 2021, doi: 10.1109/ACCESS.2021.3059858.

Using ML based systems has numerous advantages. They can be trained using large volumes of data, termed as training data, and then, through inductive inference, they can assist clinical practice in assessing risk and designing treatment.

[C]. Saif, S., Jana, M., Biswas, S. (2021). Recent Trends in IoT–Based Smart Healthcare Applying ML and DL. In: Tavares, J.M.R.S., Chakrabarti, S., Bhattacharya, A., Ghatak, S. (eds) Emerging Technologies in Data Mining and Information Security. Lecture Notes in Networks and Systems, vol 164. Springer, Singapore.

Internet of Things (IoT) is an interconnected network of different sensors, communication channels, cloud data storage, and application software. Applying machine learning algorithms in IoT-based smart healthcare systems can achieve a significant growth in prediction and detection of diseases.

[D]. Role of IoT and ML in Healthcare (K. K. . Vaigandla , Trans.). (2025). Babylonian Journal of Artificial Intelligence, 2025, 23-36. https://doi.org/10.58496/BJAI/2025/003

The growth of IoMT arrives at a time when healthcare delivery seems to be increasingly expensive. Therefore, Medtech organizations either they are start-ups or established organizations, there is a need to reinvent themselves to sustain in the competitive healthcare industry.

[E]. Venkatesh, Dr.A. Narasima, Reimagining the Future of Healthcare Industry through Internet of Medical Things (IoMT), Artificial Intelligence (AI), Machine Learning (ML), Big Data, Mobile Apps and Advanced Sensors (October 28, 2019).

The current era has witnessed the large-scale usage of cryptographic and biometric systems, and machine learning (ML) approaches for authentication and anomaly detection, respectively, for securing medical systems.

[F]. S. Pirbhulal, N. Pombo, V. Felizardo, N. Garcia, A. H. Sodhro and S. C. Mukhopadhyay, "Towards Machine Learning Enabled Security Framework for IoT-based Healthcare," 2019 13th International Conference on Sensing Technology (ICST), Sydney, NSW, Australia, 2019, pp. 1-6, doi: 10.1109/ICST46873.2019.9047745.

Using various sensors, it become possible to trace the medical health condition of the human, and a message can be forwarded to nearby hospitals which helps the patients with ease.

[G]. N. Sharma and P. G. Shambharkar, "Applicability of ML-IoT in Smart Healthcare Systems: Challenges, Solutions & Future Direction," 2022 International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, India, 2022, pp. 1-7, doi: 10.1109/ICCCI54379.2022.9740983.

Eventually this system carries out sensors such as temperature sensors, blood pressure sensors, pulse rate sensors, electrocardiography sensors and heart rate sensors where this sensor senses the information and sends all the required data to the controller.

[H]. Prabha, R., Deivanayagi, S., Kalaiselvi, V.K.G. and Rani, D.P., 2021. A review of classification algorithms in machine learning for medical IOT. International Journal of Pharmaceutical Research (09752366), 13(1).

Healthcare sectors are gradually accepting technology that provides various remote healthcare facility, disease prediction, and in-home diagnostics capabilities, which combine Machine Learning (ML) and the Internet of Things (IoT).

[I]. Agarwal, Neha, Pushpa Singh, Narendra Singh, Krishna Kant Singh, and Rohit Jain. "Machine learning applications for IoT healthcare." Machine Learning Approaches for Convergence of IoT and Blockchain (2021): 129-144.

Implanted medical devices (IMDs) and wearable devices are two examples of the kinds of IoT applications that can be used in a healthcare system based on the Internet of Things to help doctors and patients receive the best possible care.

Khang, Alex, ed. AI and IoT-based technologies for precision medicine. IGI Global, 2023.

3. ANALYSIS

3.1. Existed System

The current healthcare system largely depends on manual monitoring using standalone devices like thermometers and oximeters, which lack real-time tracking, data storage, or connectivity. These systems require human intervention for both measurement and interpretation, offering no automated alerts or AI-based diagnosis. As a result, early detection of health issues is often missed, and patients—especially the elderly or those in remote areas—face difficulty accessing timely and efficient healthcare.

3.2. Proposed System

To address the limitations of existing healthcare monitoring methods, the proposed system introduces a real-time, intelligent health monitoring solution that combines embedded systems,machine learning, and generative AI. It is designed to automate data collection, prediction, and reporting—making healthcare more proactive, accessible, and user-friendly.

- Real-time Monitoring:
  Uses ESP32 with MAX30100 and a temperature sensor to continuously collect heart rate, SpO_2, and body temperature data.
- Wireless Data Transfer:
  Sensor data is served over a local network through the ESP32's built-in web server.
- Intelligent Prediction:
  A pre-trained Random Forest ML model classifies the user's health status as either "Healthy" or "Unhealthy."
- AI-Powered Reporting:
  Google Gemini API generates a personalized health report based on vitals and predictions.
- Interactive Web Interface:
  A Streamlit web app fetches data, displays vitals in real time, shows prediction results, and presents the AI-generated report.
- PDF Download Support:
  The generated health report can be downloaded instantly for personal records or professional consultation.
- User-Friendly & Portable:
  The entire setup is lightweight, cost-effective, and can be used easily in homes, rural areas, or on the go.

3.3 HARDWARE SPECIFICATIONS

- ESP32 Development Board.
- MAX30100 Sensor
- DS18B20 Temperature Sensor
- OLED Display
- Buzzer (3–12V)
- Jumper Wires (Male–Male, Male–Female)
- Breadboard / PCB
- USB Cable (Micro-USB/Type-C)
- Power Supply (Battery/Power Bank/Adapter)
- Laptop
- Mouse

3.4. SOFTWARE SPECIFICATIONS

- Arduino IDE
- Python (3.8 or later)
- ML Model
- VS Code
- Browser (Chrome/Firefox)
- Jupyter Notebook
- Requests
- Streamlit
- Pickle
- Google Generative AI (Gemini API)
- Scikit-learn

3.5. OVERALL DESCRIPTION

- Uses ESP32 with health sensors to monitor heart rate, SpO_2, and temperature.
- Displays real-time data via a local web server.
- Streamlit web app fetches data and shows live vitals.
- ML model predicts health status as Healthy or Unhealthy.
- Google Gemini API generates an AI-based health report.
- PDF download of the report for easy access and sharing.
- A lightweight and affordable solution for remote health monitoring.

Once initiated:

- The system starts with the ESP32 microcontroller collecting real-time data from the MAX30100 (heart rate & SpO_2) and a temperature sensor.
- The ESP32 hosts a local web server to make this data accessible over Wi-Fi.
- A Streamlit-based web application fetches the sensor data from the ESP32 and displays it on an interactive dashboard.
- The data is then processed by a pre-trained Machine Learning model (Random Forest Classifier) to determine whether the health status is Healthy or Unhealthy.

If an anomaly is found:

- If the model detects abnormal values such as low SpO_2, low heart rate, or high temperature:
- A buzzer is activated to immediately alert the user.
- The OLED screen and web interface update with the detected status and current readings.
- The data, along with the model's prediction, is sent to Google Gemini API, which generates a personalized AI health report.

It ensures:

- Real-Time Alerts through buzzer and on-screen warnings.
- AI-based Reports with a one-click text file download for sharing or offline use.
- Accurate Predictions using ML for early detection.
- Data Privacy with all processing done locally—no cloud needed.
- Affordable & Portable, ideal for remote healthcare setups.
- User-Friendly Dashboard for quick understanding and accessibility.

4. SYSTEM DESIGN

The Unified Modelling Language (UML) diagrams are drawn for the project with the best possible interpretation of the project

4. UML DIAGRAMS

UML is the short form of Unified Modelling Language. UML is a standardized general-purpose modelling language in the field of object-oriented software engineering. The standard is managed, andwas created by, the Object Management Group. The important goal for UML is to create a commonmodelling language for the sake of Object-Oriented Software engineering. In its current form UMLconsists of two major components: a Meta-model and a notation. In the future, some form of method orprocess may also be added to; or associated with, UML. The Unified Modelling Language is a standardlanguage for specifying, Visualization, Constructing and documenting the artifacts of software systems, as well as for business modelling and other non- software systems. The UML represents a collection of best engineering practices that have proven successful in the modelling of large and complex systems.The UML is a very important part of developing object-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

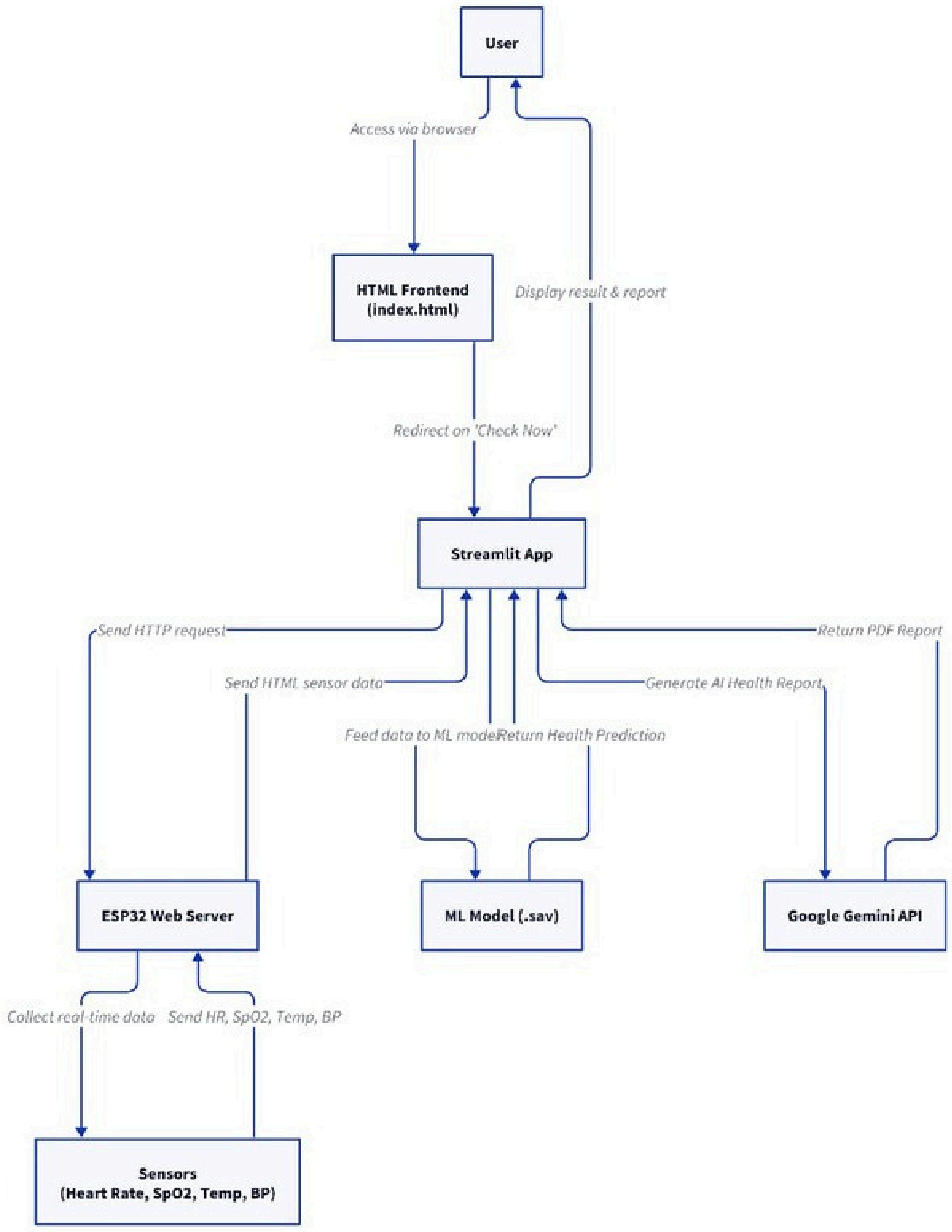
GOALS

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modelling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development processes.
4. Provide a formal basis for understanding the modelling language .
5. Encourage the growth of the Object-Oriented tools market.
6. Support higher level development concepts such as collaborations frameworks, patterns and components and Integrate best practices.
7. A UML Diagram is based on UML (Unified Modelling Language) with the purpose of visually representing a system along with its main actors, roles, actions, artifacts or classes, in order to better understand, alter, maintain, or document information about the system. The UML diagrams are divided into Structural and Behavioural UML.

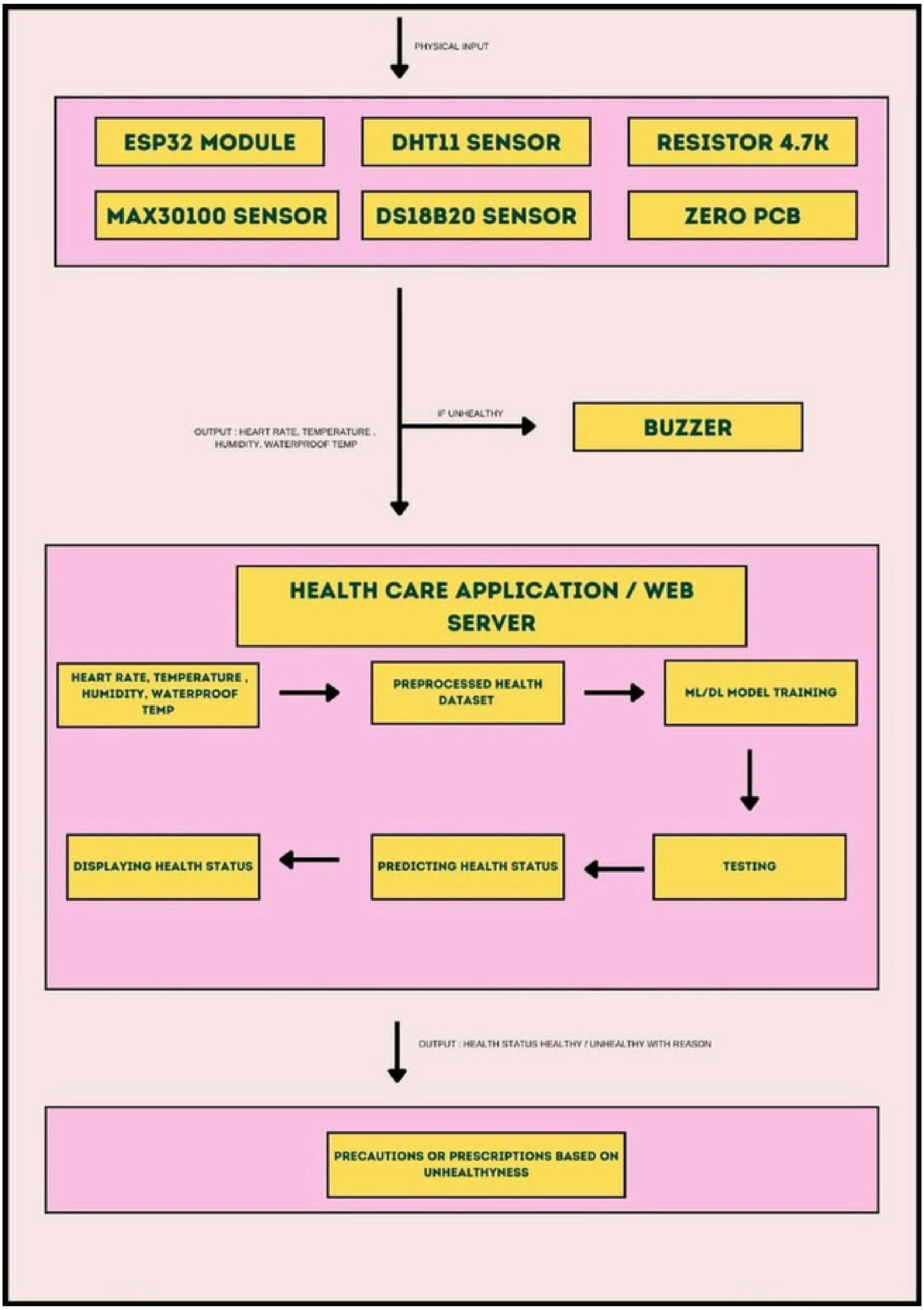
4.1. ARCHITECTURE

The architecture of a project defines the overall structure and organization of its system. It includes the arrangement of software components, their interactions, and data management. It also covers the underlying infrastructure, such as servers or cloud services, supporting the system. The architecture defines how the system meets performance, scalability, and security requirements. A well-designed architecture ensures the project is efficient, maintainable, and adaptable to future needs.



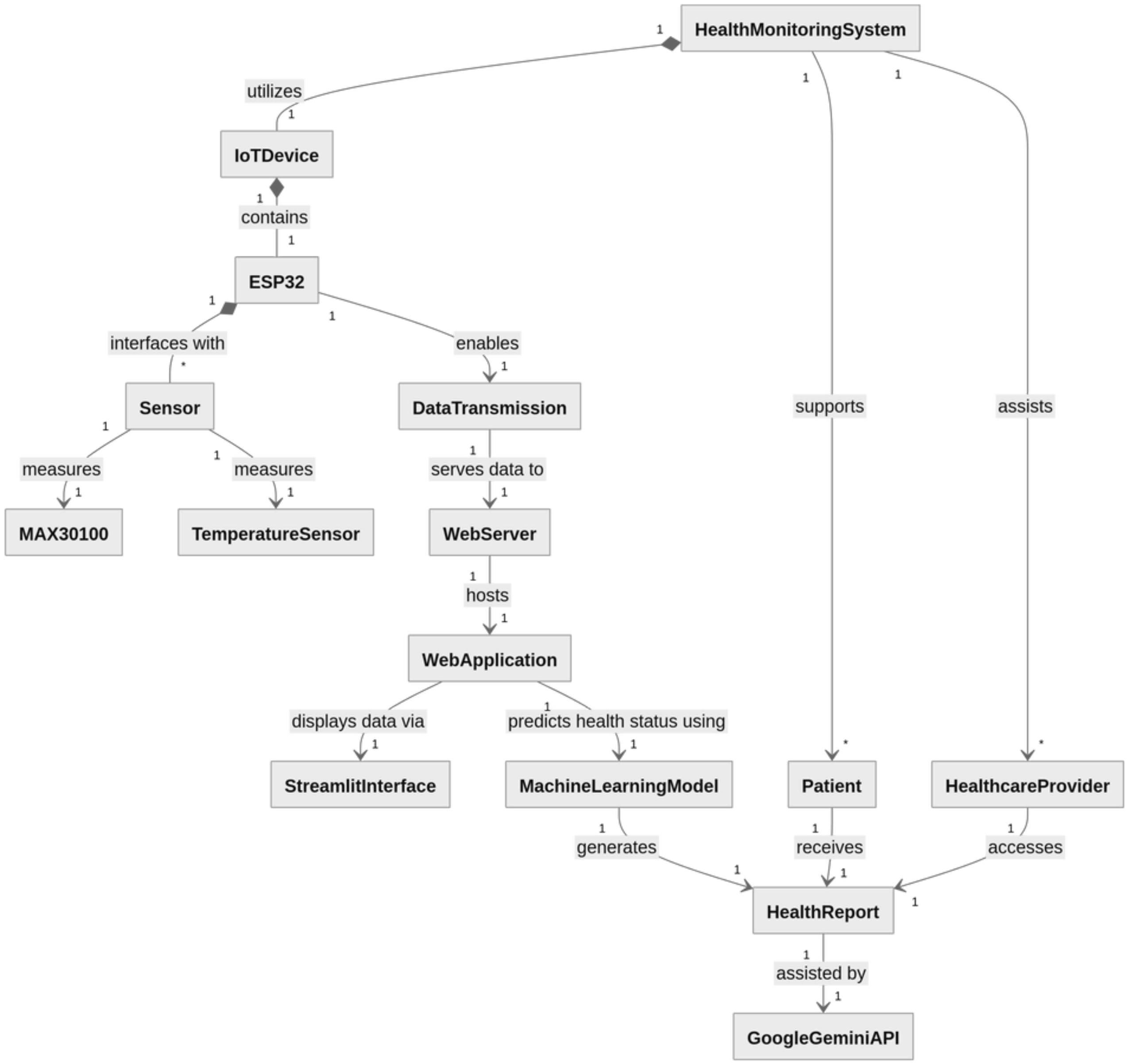
4.1.1 OBJECT DIAGRAM

A UML object diagram represents a specific instance of a class diagram at a certain moment in time.When represented visually, you'll see many similarities to the class diagram. An object diagram focuses on the attributes of a set of objects and how those objects relate to each other


PHYSICAL INPUT
ESP32 MODULE
DHT11 SENSOR
RESISTOR 4.7K
MAX30100 SENSOR
DS18B20 SENSOR
ZERO PCB
OUTPUT : HEART RATE, TEMPERATURE , HUMIDITY, WATERPROOF TEMP
IF UNHEALTHY
BUZZER
HEALTH CARE APPLICATION / WEB SERVER
HEART RATE, TEMPERATURE , HUMIDITY, WATERPROOF TEMP
PREPROCESSED HEALTH DATASET
ML/DL MODEL TRAINING
TESTING
PREDICTING HEALTH STATUS
DISPLAYING HEALTH STATUS
OUTPUT : HEALTH STATUS HEALTHY / UNHEALTHY WITH REASON
PRECAUTIONS OR PRESCRIPTIONS BASED ON UNHEALTHYNESS

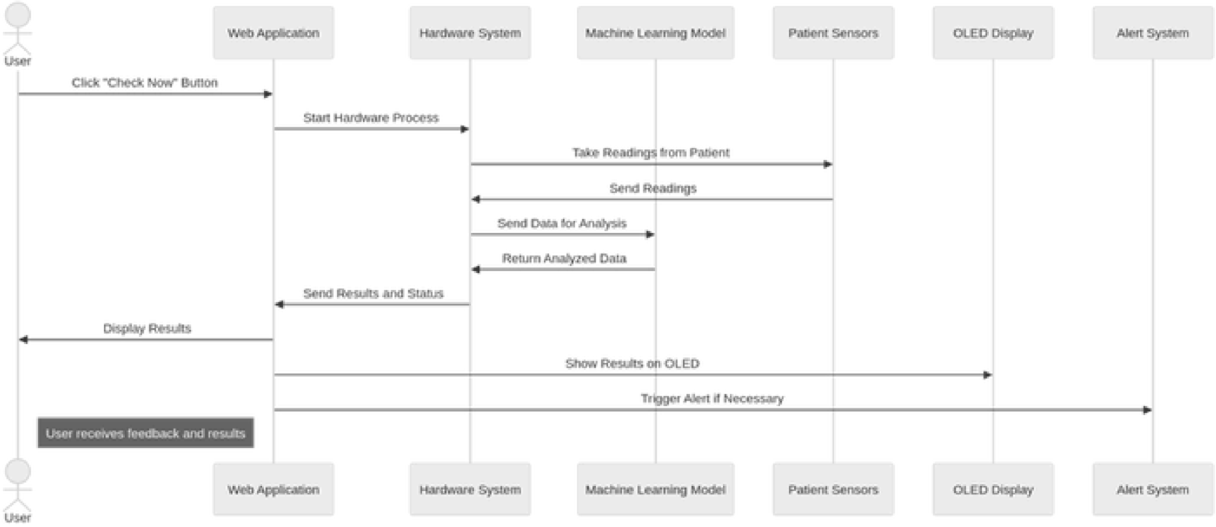
4.1.2. CLASS DIAGRAM

A class diagram is a type of static structure diagram in software engineering that illustrates the structure of a system by depicting its classes, attributes, operations (methods), and the relationships among objects. It provides a visual representation of the classes and their associations within the system, serving as a blueprint for the software design and architecture.



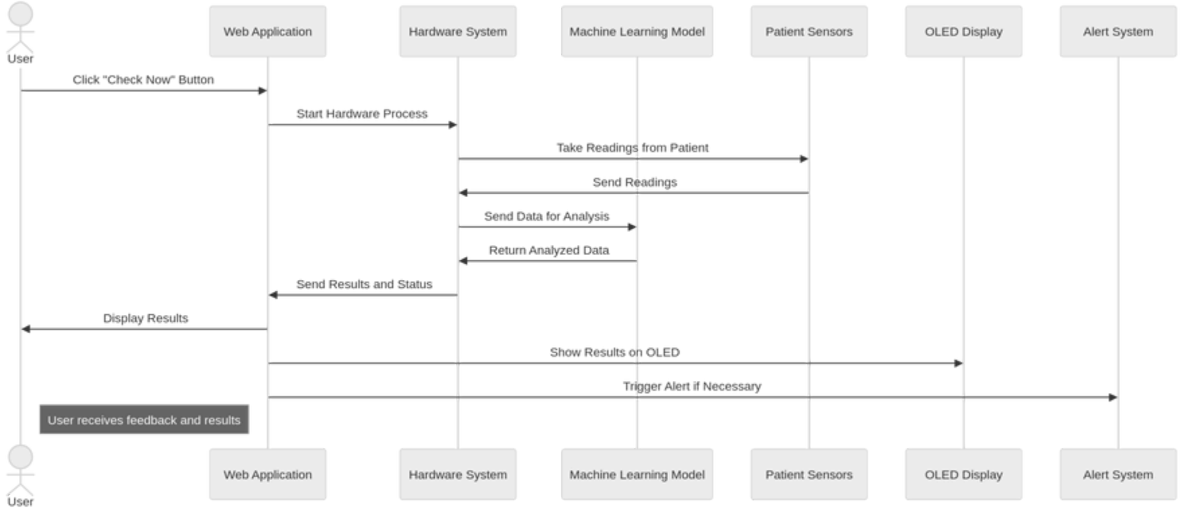
4.1.3. SEQUENCE DIAGRAM

A sequence diagram in UML illustrates how objects interact with each other over time, showing the sequence of messages exchanged between them. It emphasizes the order of events or actions in a specific scenario or process.



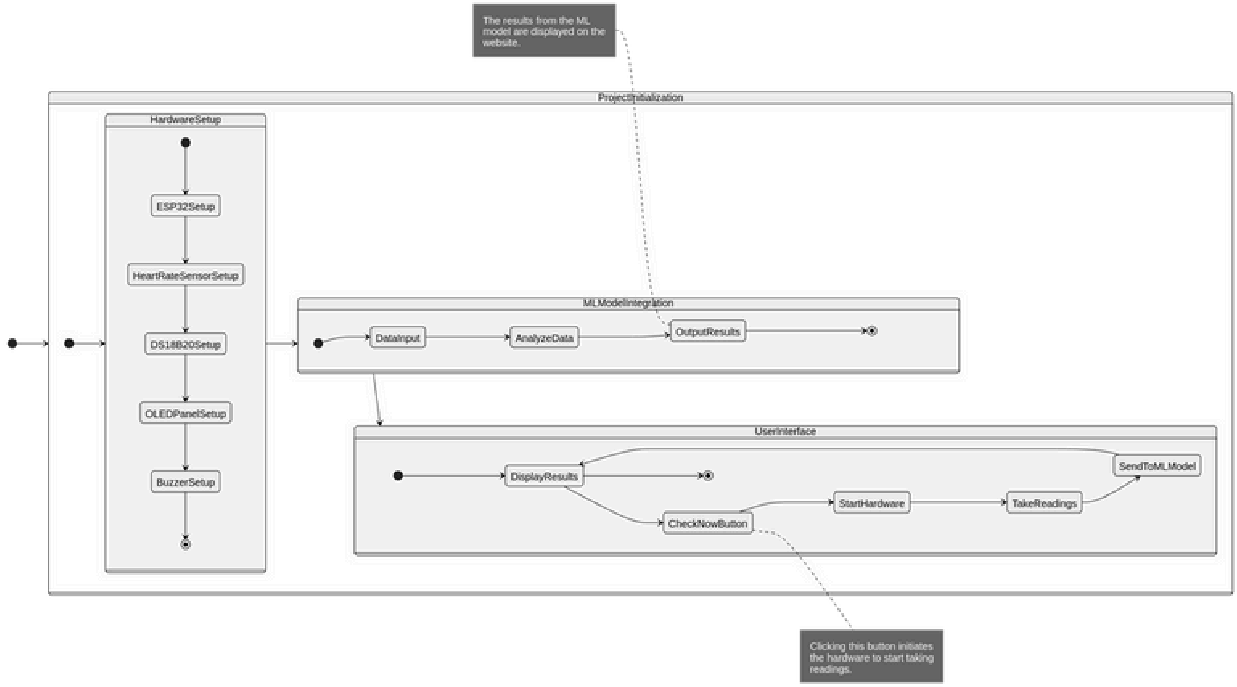
4.1.4 ACTIVITY DIAGRAM

An activity diagram in UML models the flow of activities or tasks within a system, showing the sequence of actions, decisions, and concurrency. It is used to represent workflows, business processes, or use case scenarios.



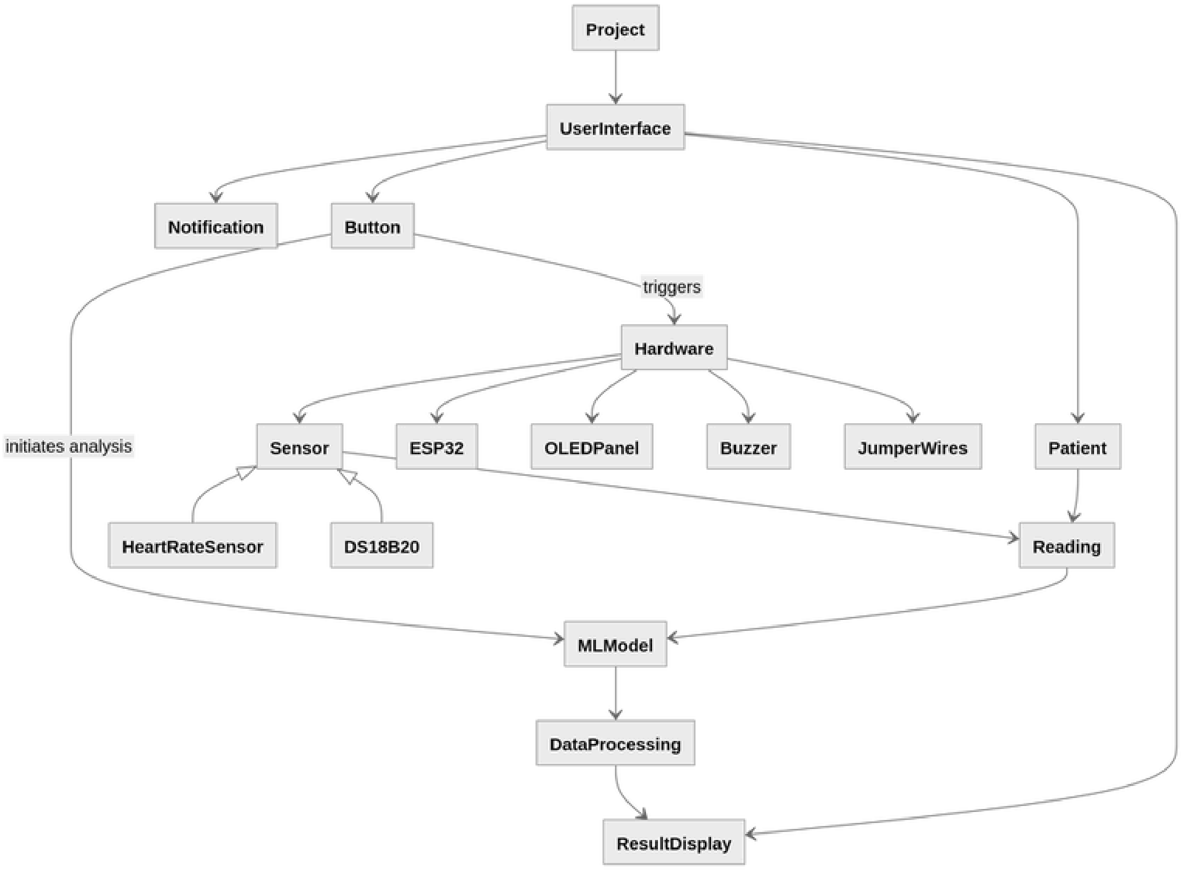
4.1.5. STATE MACHINE DIAGRAM

A state machine diagram in UML is a visual representation of an object's behavior, showcasing its states and transitions. It illustrates how events trigger state changes, often with conditions (guards) and actions, providing a clear understanding of complex systems

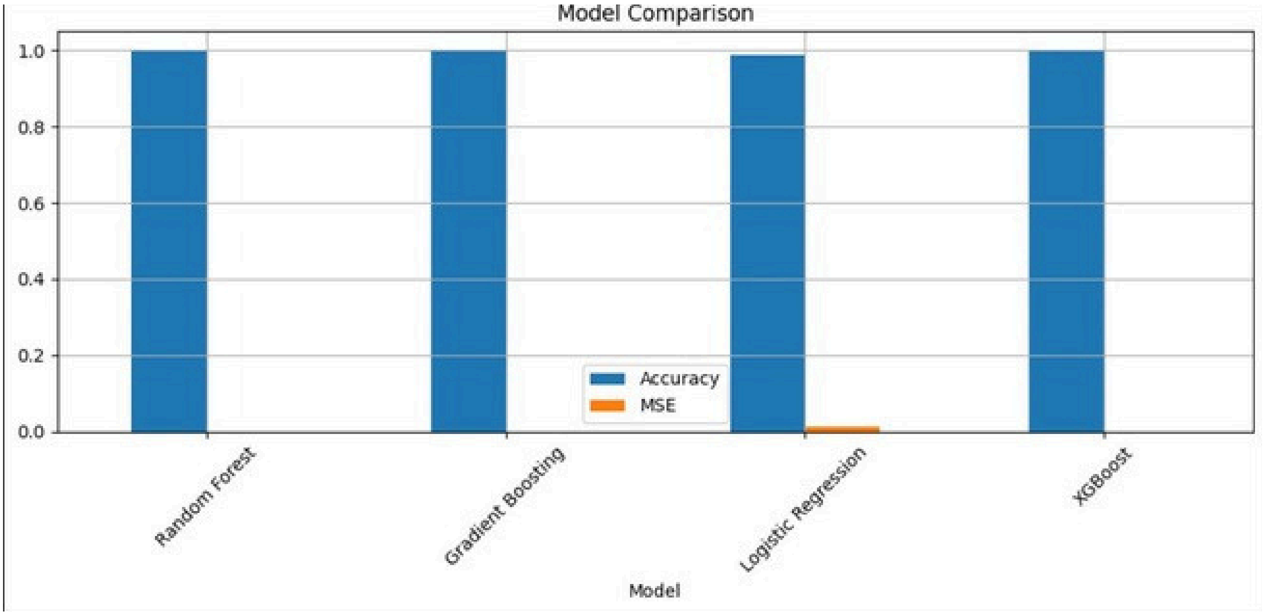


4.1.6. COMPONENT DIAGRAM

A component diagram in UML shows the structure of a system by representing its components e.g., software modules or hardware devices) and how they interact with each other. It focuses on the organization and dependencies between components



4.2. STATISTICAL ANALYSIS



Model Comparison: X-Axis:

Different ML models

- Random Forest
- Gradient Boosting
- Logistic Regression
- XGBoost

Y-Axis: Performance Metric Values (range: 0 to 1)

Legend:

- Blue bars = Accuracy (how often the model correctly predicted the class). Orange
- bars = MSE (Mean Squared Error — lower is better; ideal is close to 0).

Detailed Analysis:

- Model Accuracy (Blue Bar) MSE (Orange Bar) Interpretation.
- Random Forest ~1.0 ~0.0 Perfect accuracy; no errors.
- Gradient Boosting ~1.0 ~0.0 Excellent performance.
- Logistic Regression Slightly <1.0 Slightly >0.0 Still very good, but not perfect.
- XGBoost ~1.0 ~0.0 Top-tier result.

Key Takeaways: All models performed extremely well, especially ensemble methods (Random Forest, Gradient Boosting, XGBoost), achieving nearly 100% accuracy and almost zero error (MSE).

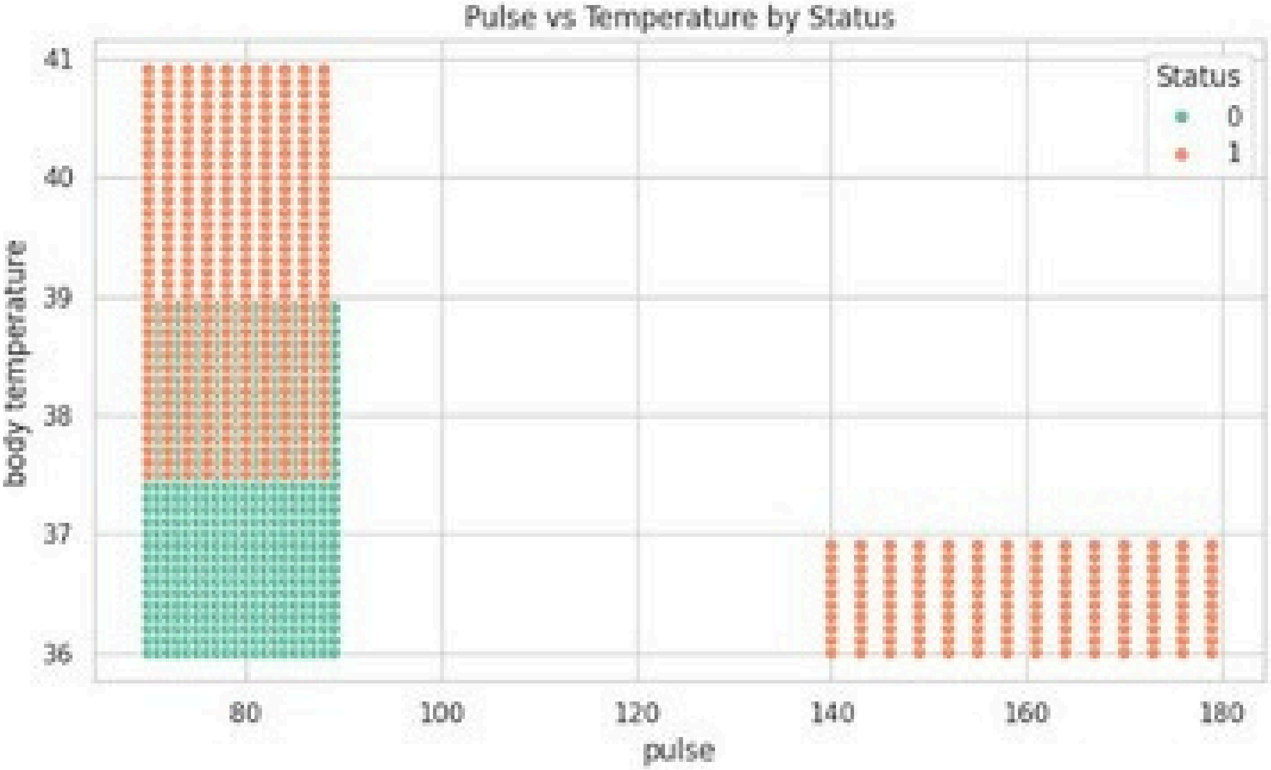
- Logistic Regression, while still solid, had slightly lower accuracy and slightly higher error. This is expected, as it's a simpler linear model compared to the powerful non- linear ensembles.

Why Ensemble Models Performed Better?

- Model Strength.
- Random Forest Reduces overfitting by averaging many decision trees.
- Gradient Boosting Builds trees sequentially to fix errors, improving precision.
- XGBoost Highly optimized and regularized boosting model, often state-of-the-art.
- Logistic Regression Assumes linearity; might miss complex patterns in health data.

Recommendations:

- For deployment in your health monitoring project, you can confidently choose:
- XGBoost or Random Forest.
- They are accurate, stable, and great with mixed data types.
- Use Logistic Regression as a lightweight, interpretable backup or baseline.



Plot 1: Pulse vs Body Temperature by Status

What it shows:

- This plot maps pulse on the x-axis and body temperature on the y-axis, with points colored by Status:
- Green dots (Status = 0 → Healthy).
- Orange dots (Status = 1 → Unhealthy).

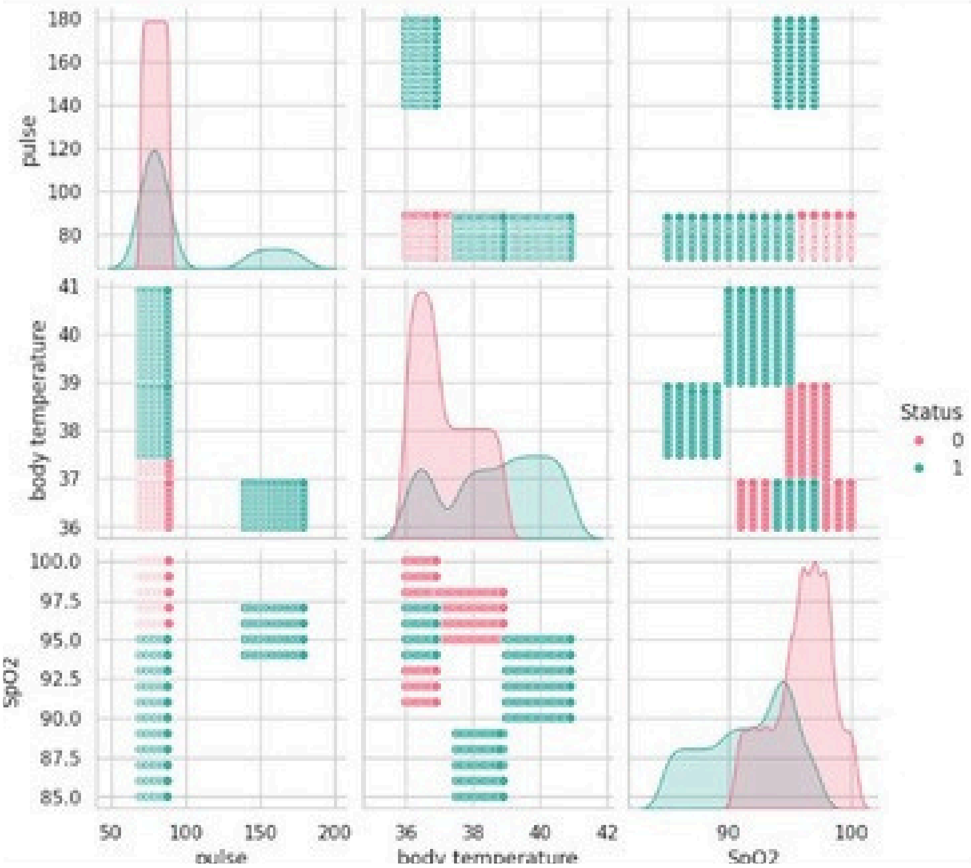
Observations:

- Feature Healthy (0) Unhealthy (1).
- Pulse 72–90 bpm (tight cluster) 75–180 bpm (wider spread).
- Temperature Mostly 36–38.5°C Mostly 38–41°C.
- Healthy individuals:
- Clustered tightly between pulse 75–85 bpm and body temp 36–38.5°C.
- No extreme pulse or temperature.

Unhealthy individuals:

- Pulse range is more varied, going up to 180 bpm.
- Body temperature goes up to 41°C, clearly feverish.
- Some patients have high pulse + high temperature, typical signs of stress or feverish conditions.

Insight: High body temperature paired with high or erratic pulse is strongly associated with illness.



Top Left to Bottom Right – Diagonal Elements (Feature Distributions)

pulse:

- Status 0 (pink): Highly concentrated between 80–100 bpm.
- Status 1 (teal): More spread out with a cluster around 120–140 bpm.
- Indicates high pulse values are associated with "unhealthy" status.

Body temperature:

- Status 0: Peaks around 36.5°C–37°C.
- Status 1: Distribution skews toward higher values (38°C+).
- Fever (38°C+) is common in unhealthy individuals.

- **SpO2:**
- Status 0: Most values are 96%–99%.
- Status 1: Drops towards 85%–95%.
- Low oxygen saturation (<95%) is linked to unhealthy status.

Off-Diagonal Elements – Pairwise Scatter Plots

Each off-diagonal scatter plot compares two features with points colored by Status. Here's what each tells us:

.

pulse vs body temperature (top row, second column)

- Healthy (pink) samples are clustered in a tight rectangle: pulse 80–100, temp 36–37.5.
- Unhealthy (teal) samples are more scattered, generally with:
- Higher temperature.
- Higher or lower pulse.

Suggests that elevated temperature + abnormal pulse is a red flag.

- pulse vs SpO2 (top row, third column)
- Healthy (pink) cluster: pulse < 100, SpO2 > 95%.

Unhealthy (teal): Often have either:

- High pulse (>110) or
- Lower SpO2 (<94%)
- Combined deviation in SpO2 and pulse correlates with poor health.

Temperature vs SpO2 (middle row, third column)

- Healthy (pink) cluster in: temp 36–37.5, SpO2 > 95%

Unhealthy (teal):

- Tend toward higher temp (>38°C) and lower SpO2 (<93%)
- Important biomarkers for illness: fever + low oxygen saturation.

Color Legend (hue)

- Status 0: Healthy (pink)
- Status 1: Unhealthy (teal)

This helps visually segment healthy/unhealthy cases in each plot.

Takeaways for Modeling & Feature Engineering

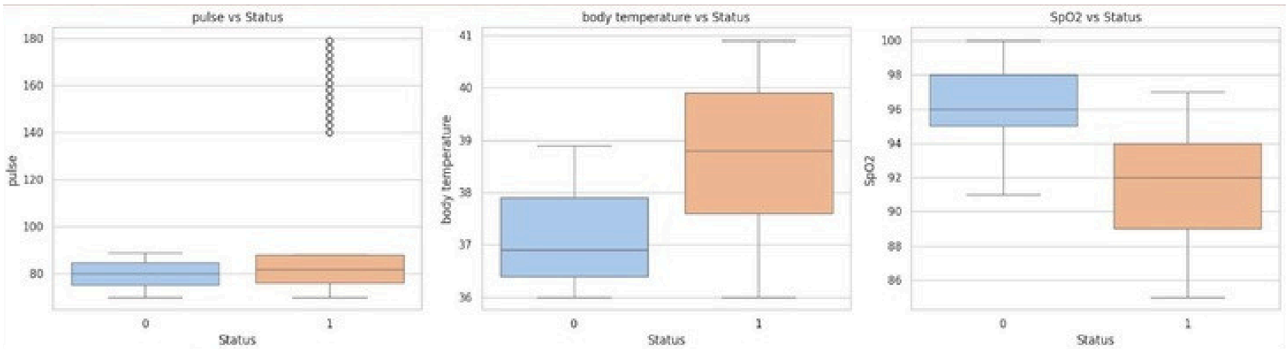
These plots visually confirm that pulse, body temperature, and SpO2 are discriminative features.

For example:

High pulse + low SpO2 = likely unhealthy.

Normal temp + high SpO2 = likely healthy.

These relationships are nonlinear and multivariate, so models like Random Forest, XGBoost, or SVM can capture them better than linear ones.



Seaborn Boxplots:

Seaborn Boxplots to visualize the distribution of three key physiological features (pulse, body temperature, SpO2) across the binary target class Status:

- Status = 0: Healthy (blue)
- Status = 1: Unhealthy (orange)

Boxplot:

- A boxplot (or box-and-whisker plot) shows:
- Box: Interquartile range (IQR = Q3 - Q1), where 50% of the data lies.
- Line inside the box: Median (Q2).
- Whiskers: Extends to data within 1.5×IQR from Q1 and Q3.
- Dots outside whiskers: Outliers.

This is super useful for comparing distributions and spotting differences between groups.

Plot-by-Plot Breakdown

1. Pulse vs Status

- Healthy (0) Unhealthy (1)
- Median ≈ 80 Median ≈ 88
- **Range: 72–87 Range: 75–175 (many outliers)**

Observations:

- Healthy individuals tend to have lower pulse rates, tightly clustered.
- Unhealthy individuals have:
- Slightly higher median.
- More variability.
- Many outliers, some with pulse >160 bpm.

High pulse variability is a marker of illness.

2. Body temperature vs Status

- Healthy (0) Unhealthy (1)
- Median ≈ 37.0 °C Median ≈ 39.0 °C
- Range: 36–38.8 Range: 37–41

Observations:

- Healthy individuals have normal body temperature.
- Unhealthy individuals show: Fever (≥38°C) in
- majority cases. Higher upper whiskers and outliers
- around 41°C. Elevated temperature is a strong
- indicator of illness.

3. SpO2 vs Status

- Healthy (0) Unhealthy (1)
- Median ≈ 97% Median ≈ 92%
- Range: 95–100% Range: 85–96%

Observations:

- Healthy individuals maintain normal oxygen saturation (above 95%).
- Unhealthy individuals:
- Median drops to 92%.
- Minimum dips to 85%.
- Low SpO2 is clinically significant, especially in respiratory issues or infections like COVID-19.

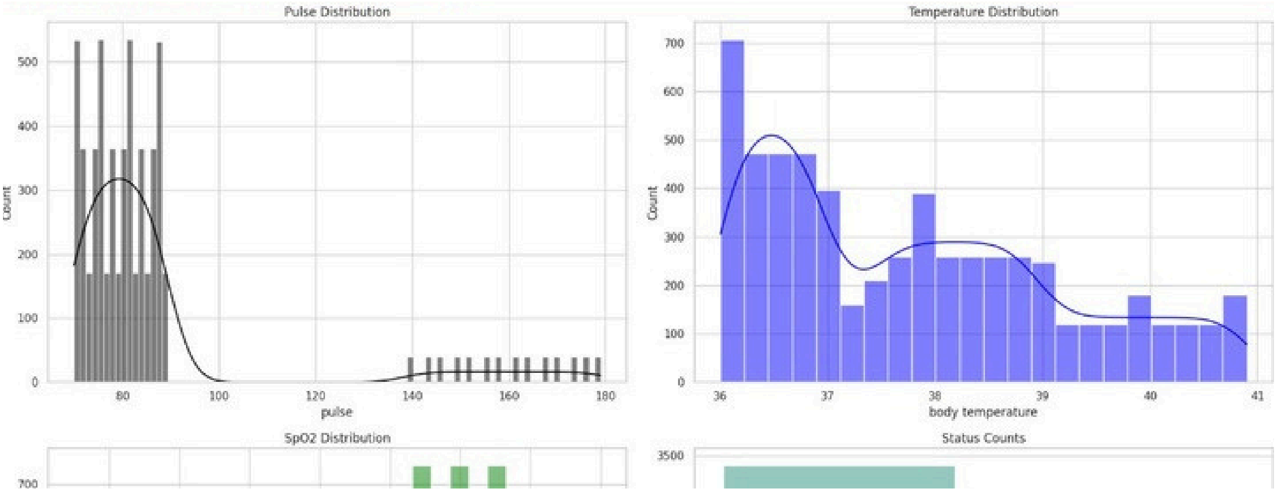
Supporting Scatter Plot (Code Below the Boxplots)

You're also generating a scatter plot of:

- python
- Copy
- Edit
- x='pulse', y='body temperature', hue='Status'
- This is meant to show how these two features interact based on health status.

Conclusion from All 3 Boxplots

- Feature Healthy (Status 0) Unhealthy (Status 1) Insight.
- Pulse 72–87, stable 75–175, variable High variability = illness.
- Body Temperature 36–38 °C 37–41 °C Fever clearly separates classes.
- SpO2 95–100% 85–96% Low oxygen = illness.
- These plots strongly justify your use of these three features in your health prediction ML model.



Top Left: Pulse Distribution

- This plot shows a histogram of the pulse variable.
- Most of the pulse values are tightly clustered between 70 and 100 bpm, which is the normal resting heart rate range.
- There’s a small tail toward higher pulse values (120–180 bpm), indicating a few people with abnormally high pulse — potentially unhealthy cases.
- The black KDE (kernel density estimate) curve overlays a smoothed approximation of the distribution shape.

Interpretation:

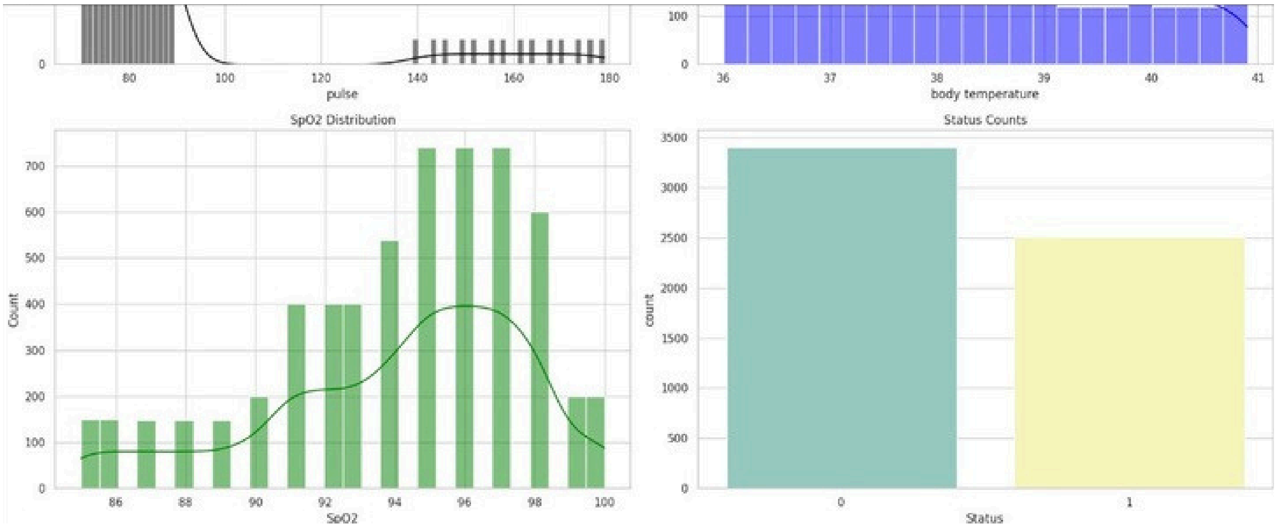
- Most individuals have normal pulse.
- Some outliers (right tail) suggest elevated heart rate — which might be contributing to health risks.

Top Right: Temperature Distribution

- A histogram of body temperature, with a blue KDE overlay.
- The temperatures peak between 36°C and 37.5°C, which is within normal body temperature range.
- There’s a visible second hump beyond 38°C, indicating a secondary group — possibly people with fever or health issues.
- A few extreme values go beyond 40°C.

Interpretation:

- The bimodal distribution suggests two clear groups: healthy and febrile.
- Elevated temperature (>38°C) often correlates with the "Unhealthy" label.



Bottom Left: SpO2 Distribution

- This histogram shows SpO2 (oxygen saturation). Most values lie between 92%
- and 98%, which is the normal range. No KDE line is shown here, but the bars
- are tightly packed with little skewness.

Interpretation:

- Individuals mostly have healthy oxygen levels. If there are any below 90%, that's a red
- flag and might contribute to an unhealthy classification.

Bottom Right: Status Counts

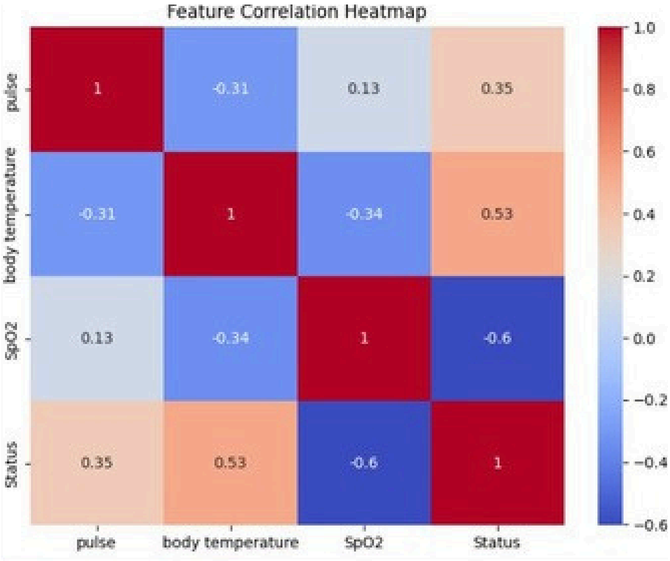
- This is a countplot of the Status variable. It shows the frequency of each class
- — probably "Healthy" vs "Unhealthy". One of the bars is significantly taller —
- likely indicating more healthy samples.

Interpretation:

- You may have an imbalanced dataset, where one class dominates. It's important to
- handle this imbalance using resampling techniques (like SMOTE or class weights) during
  model training.

Summary: Why This Is Useful

- These distribution plots help visualize skewness, outliers, and group patterns. They give
- insight into how different features might influence health status, e.g., fever, low SpO2,
  and high pulse being associated with poor health. Helps guide feature selection and
- preprocessing for your ML model.



Correlation Heatmap

A correlation matrix displays how closely related two variables are. It uses Pearson correlation coefficients which range from:

- +1 → Perfect positive correlation (as one increases, the other increases)
- 0 → No correlation
- −1 → Perfect negative correlation (as one increases, the other decreases)

In the heatmap:

- Redder square mean higher positive correlation.
- Bluer squares mean higher negative correlation.
- White or near-zero values mean no significant relationship.

Interpretation of the Heatmap

- Feature Pair Correlation Interpretation.
- pulse vs Status 0.35 Moderate positive: higher pulse → more likely to be "Unhealthy".
- body temperature vs Status 0.53 Stronger positive: higher body temp → more likely to be "Unhealthy".

Body temperature vs Status 0.53 Stronger positive: higher body temp → more likely to be "Unhealthy"

SpO2 vs Status -0.6 Strong negative: higher SpO2 → more likely to be "Healthy".

- pulse vs body temperature -0.31 Weak negative: not strongly related.
- pulse vs SpO2 0.13 Very weak positive.
- body temperature vs SpO2 -0.34 Moderate negative: higher temp might mean lower
- SpO2.

Key Insights

- SpO2 is inversely correlated with poor health. A low SpO2 is a strong indicator of being unhealthy.

- Body Temperature has a strong positive correlation with being unhealthy.

- Pulse shows some relationship but not as strong as the others.

5. IMPLEMENTATION

5.1 MODULES

1. Sensor Module

- MAX30100 Sensor – For measuring heart rate and SpO_2 (oxygen saturation).
- Temperature Sensor (e.g., DS18B20 or DHT11) – For measuring body temperature.

2. Microcontroller Module

- ESP32 – Collects data from sensors and hosts a local web server to make the data accessible over Wi-Fi.

3. Web Interface Module

- Streamlit Web Application – Fetches real-time data from the ESP32 and displays it on an interactive dashboard.

4. Machine Learning Module

- Random Forest Classifier (Pre-trained ML Model) – Predicts the user's health status as Healthy or Unhealthy based on sensor inputs.

5. AI Reporting Module

- Google Gemini API – Generates a personalized AI health report based on live data and ML prediction.

6. Text file Generation Module

- Allows the AI-generated report to be downloaded as a text file for easy sharing and record keeping.

7. Connectivity & Hosting Module

- ESP32 Local Web Server – Hosts the data locally, no internet required, making it suitable for remote use.

8. User Feedback Module

- Buzzer – Gives an instant alert if an anomaly is detected. OLED Display –
- Displays current vitals and health status directly on the device.

5.2 MODULE DESCRIPTION

1. Sensor Module This module is responsible for continuously acquiring health-related data from the user in real time. It consists of:

- MAX30100 Sensor: A combination pulse oximeter and heart rate sensor that uses photodetectors and LEDs to measure SpO_2 (oxygen saturation) and pulse rate.
- Temperature Sensor (DHT11 or DS18B20): Monitors body temperature. DS18B20 is more accurate and digital, while DHT11 is more basic but cost-effective.

These sensors are the first point of contact for data collection and play a critical role in ensuring reliable health monitoring.

2. Microcontroller Module (ESP32)

- The ESP32 microcontroller acts as the brain of the system. It interfaces directly with the connected sensors, processes raw data, and initiates further communication.
- It is Wi-Fi-enabled, allowing it to host a local web server and transfer sensor readings wirelessly.

3. Web Server Module

- The ESP32 runs a local web server that presents the sensor readings (heart rate, SpO_2, and temperature) in a simple HTML format.
- This allows the sensor data to be accessed from any device connected to the same Wi-Fi network, such as laptops or smartphones, making the system wireless and convenient.

4. Frontend Visualization Module (Streamlit Web App)

- A Streamlit-based web application is developed to create a clean and interactive UI that fetches live data from the ESP32 web server. It displays the vitals clearly using charts or
- text, shows the output of the ML model (Healthy or Unhealthy), and provides real-time updates.

5. Machine Learning Module

- A pre-trained Random Forest Classifier is used to analyze the input vitals and predict health status as either Healthy or Unhealthy. It uses ensemble learning to reduce
- overfitting and improve accuracy. The model is hosted within the Streamlit app (or a
- Flask backend) and triggered each time new data is fetched from the ESP32.

6. Generative AI Report Module

- This module utilizes the Google Gemini API, a powerful generative AI model, to generate a natural language health report. The AI uses input from the sensor values and
- ML prediction to write a personalized report, including basic recommendations, interpretation of the data, and health advice. This makes the system informative and
- user-friendly, especially for non-technical or non- medical users.

7. PDF Report Module

- Once the AI report is generated, it is passed to a text file conversion tool to enable a downloadable health report. This allows patients or users to save the report for
- medical reference or share it with doctors or family members.

8. User Alert Module

- The system includes a buzzer that provides immediate sound alerts if any anomaly is detected in the vitals (e.g., high temperature, low oxygen, abnormal heart rate). An OLED
- Display is also integrated into the ESP32 setup, showing real-time values and prediction status, even without accessing the web app.

5.3 TECHNOLOGIES USED

1. Embedded Systems & Hardware

- ESP32 Microcontroller – For sensor interfacing, data processing, and hosting the local web server.
- MAX30100 Sensor – Measures heart rate and oxygen saturation (SpO_2).
- DS18B20 or DHT11 – For body temperature monitoring.
- OLED Display (1.3 inch) – Displays real-time vitals and system messages. Buzzer – Provides instant alerts on anomaly detection.
- Jumper Wires, Breadboard/PCB – For connections and setup.

2. Programming Languages

- MicroPython / C++ (Arduino IDE) – To program ESP32 for sensor control and web server setup. Python 3.x – For backend, machine learning, AI integration, and
- frontend (Streamlit).

3. Web & UI Technologies

- HTML/CSS – For ESP32-hosted web server UI displaying raw sensor data.
- Streamlit – For creating an interactive and dynamic health monitoring dashboard in Python.

4. Machine Learning

- scikit-learn – For building and using the Random Forest Classifier ML model to predict health status.
- Pickle – To save and load the trained ML model.

5. Generative AI

- Google Gemini API (via google.generativeai library) – To generate personalized, AI- written health reports based on predictions and vitals.

6. Networking & Communication

- ESP32 Wi-Fi – Used for local server hosting and wireless data transmission. HTTP
- Requests (Python requests library) – Used by Streamlit app to fetch sensor data from ESP32.

7. Report Generation

- ReportLab – To convert the AI-generated health report into a downloadable text file format

5.4 SAMPLE CODE

1.Sample HTML code

```
<!DOCTYPE html>
<html lang="en">
<head>
<link rel="preconnect" href="https://fonts.googleapis.com">
<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
<link rel="preconnect" href="https://fonts.googleapis.com">
<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
<link href="https://fonts.googleapis.com/css2?
family=Poppins:ital,wght@0,100;0,200;0,300;0,400;0,500;0,600;0,700;0,800;0,900;1,100;1,200;1,3
00;1,400;1,500;1,600;1,700;1,800;1,900&family=Roboto:ital,wght@0,100..900;1,100..900&display
=swap" rel="stylesheet">
<link href="https://fonts.googleapis.com/css2?
family=Poppins:ital,wght@0,100;0,200;0,300;0,400;0,500;0,600;0,700;0,800;0,900;1,100;1,200;1,3
00;1,400;1,500;1,600;1,700;1,800;1,900&display=swap" rel="stylesheet">
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>MediCare+ | AI Health Monitoring</title>
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/6.4.0/css/all.min.css">
<link rel="stylesheet" href="styles.css">
</head>
<body>
<!-- Navigation Bar -->
<header class="header">
<a href="#" class="logo"><i class="fas fa-heartbeat"></i> MediCare+</a>
<nav class="navbar">
<a href="#home">Home</a>
<a href="#services">Services</a>
<a href="#team">Team</a>
<a href="diagnose.html">Diagnose</a>
<a href="#contact">Contact</a>
</nav>
<div id="menu-btn" class="fas fa-bars"></div>
</header>
```

2.Sample Python code

```
import streamlit as st
import numpy as np
import pickle
import requests
from bs4 import BeautifulSoup
from io import BytesIO
import google.generativeai as genai

# Load ML model
with open("rfmodel.sav", "rb") as model_file:
model = pickle.load(model_file)

# Configure Gemini API
genai.configure(api_key="AIzaSyAHsBQL8k1dfejBFVd7AMIECr3F6m4xPPY")
model_gemini = genai.GenerativeModel("gemini-1.5-pro")
# ESP32 server URL
ESP32_URL = "http://192.168.249.163/"
# Set Streamlit page config
st.set_page_config(page_title="Health Status Predictor", layout="centered")
st.title("🩺 Real-Time Health Status Prediction + AI Report")
# Function to fetch and parse ESP32 sensor values
def fetch_esp32_data():
try:
res = requests.get(ESP32_URL, timeout=10)
soup = BeautifulSoup(res.text, "html.parser")
# Parse each line
paragraphs = soup.find_all("p")
temp = float(paragraphs[0].text.split(":")[1].replace("°C", "").strip().replace('"', ''))
pulse = int(paragraphs[1].text.split(":")[1].replace("BPM", "").strip().replace('"', ''))
spo2 = int(paragraphs[2].text.split(":")[1].replace("%", "").strip().replace('"', ''))


 return pulse, temp, spo2
```

3. Sample Java Script code

```
document.addEventListener("DOMContentLoaded", function () {
const checkNowButton = document.getElementById("check-now");
const statusText = document.getElementById("status-text");
const resultText = document.getElementById("result-text");
checkNowButton.addEventListener("click", async function () {
statusText.textContent = "Starting hardware check...";
resultText.textContent = "";
try {
// Step 1: Request ESP32 to start measurement
let espResponse = await fetch("http://192.168.4.1/start"); // Replace with your ESP32 IP
let sensorData = await espResponse.json(); // Get the heart rate & temperature
if (!sensorData.heartRate || !sensorData.temperature) {
throw new Error("Invalid sensor readings from ESP32.");
}
console.log("Sensor Data from ESP32:", sensorData);
statusText.textContent = "Processing data...";
// Step 2: Send sensor data to ML model for diagnosis
let mlResponse = await fetch("http://localhost:5000/predict", { // Replace with your ML API URL
method: "POST",
headers: { "Content-Type": "application/json" },
body: JSON.stringify(sensorData)
});
let mlResult = await mlResponse.json(); // Get ML model output
console.log("ML Prediction:", mlResult);
// Step 3: Display result on the UI
statusText.textContent = "Diagnosis complete.";
resultText.textContent = `Health Status: ${mlResult.status}`; // Display "Healthy" or "Unhealthy"
} catch (error) {
console.error("Error:", error);
statusText.textContent = "Error: " + error.message;
} });
});
```

4. Sample CSS code

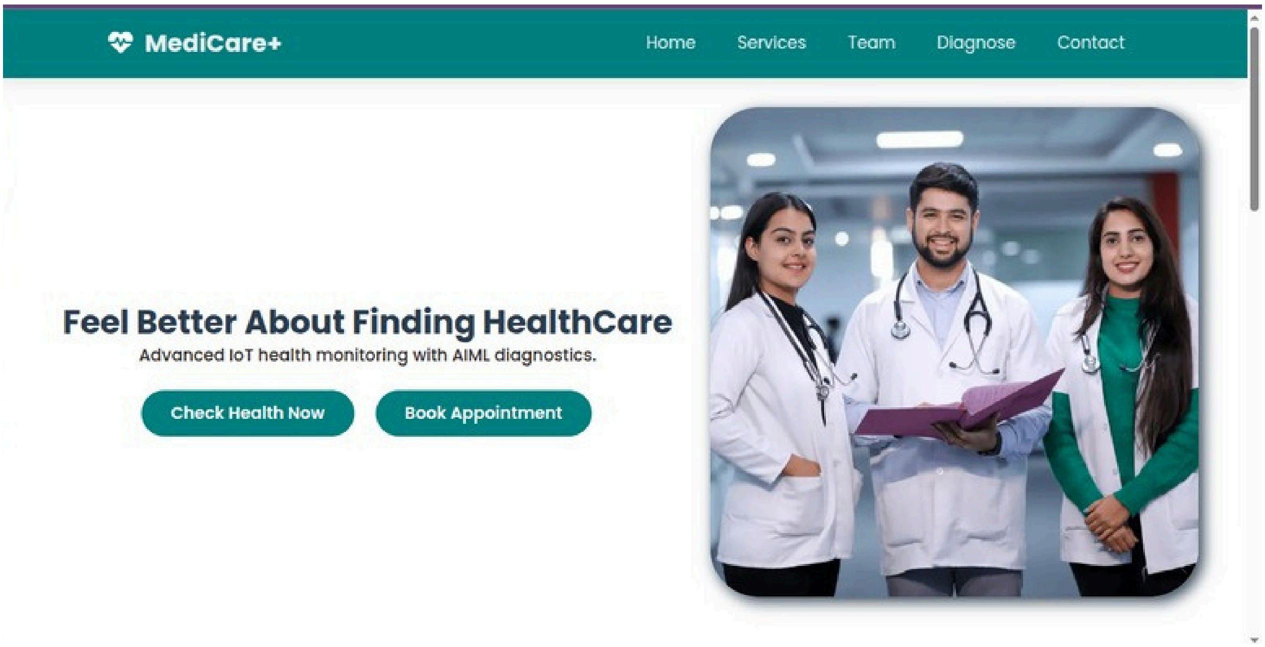
```
/* ===== Base Variables ===== */
:root {  primary: #16a085;
    primary-dark: #138a76;
secondary: #2c3e50;
accent: #3498db;
success: #2ecc71;
danger: #f7f5f4;
light: #eef2f3;
gray: #95a5a6;
white: #ffffff;
nav-bg: #2c3e50;
nav-text: #ecf0f1;
shadow: 0 4px 20px rgba(0, 0, 0, 0.1);

 /* --gradient: linear-gradient(135deg, #16a085 0%, #3498db 50%, #2ecc71 100%); */
}
.home{
background: rgb(250, 246, 246);
padding: 5rem;
border-radius: 1rem;
box-shadow: var(--shadow);
font: poppins;}
.contact {
background: white;
padding: 5rem;
border-radius: 1rem;
box-shadow: var(--shadow);
font: poppins;}
.team {
background: rgb(247, 241, 241);
padding: 5rem;
border-radius: 1rem;
box-shadow: var(--shadow);
font: poppins;}
```

5. Sample Hardware code

```
#include <WiFi.h> #include <Wire.h>
#include "MAX30100_PulseOximeter.h"
#include <WebServer.h> #include
<OneWire.h> #include
<DallasTemperature.h> #include
<Adafruit_GFX.h> #include
<Adafruit_SSD1306.h>


// Wi-Fi Credentials
const char* ssid = "Your_SSID";
const char* password = "Your_PASSWORD";

// MAX30100 Pins
#define MAX30100_SDA 33
#define MAX30100_SCL 32

// DS18B20 Pin
#define ONE_WIRE_BUS 5

// Buzzer Pin
#define BUZZER_PIN 27

// OLED Display Settings #define SCREEN_WIDTH 128 #define
SCREEN_HEIGHT 64 Adafruit_SSD1306 display(SCREEN_WIDTH,
SCREEN_HEIGHT, &Wire, -1);



#define REPORTING_PERIOD_MS 60000 // Update sensor data every 60
seconds


PulseOximeter pox;
WebServer server(80);
OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature sensors(&oneWire);
```

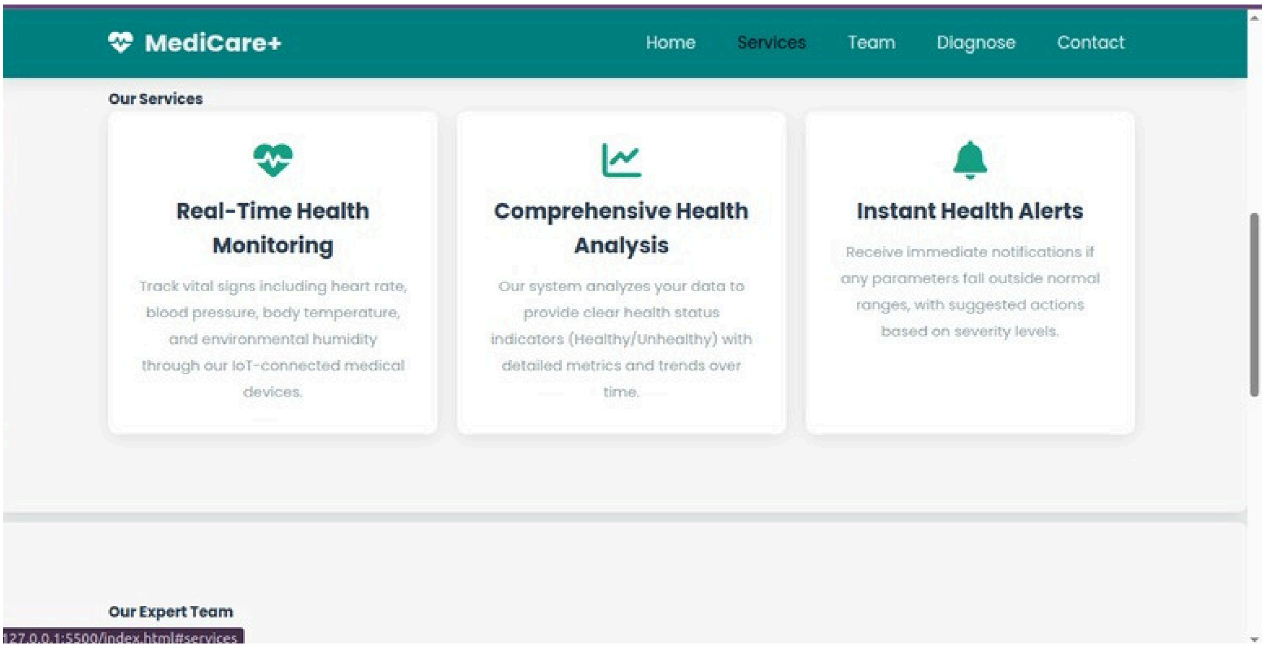
```
uint32_t tsLastReport = 0;
float heartRate = 0.0, spO2 = 0.0, temperatureC = 0.0;

// Callback for heartbeat detection
void onBeatDetected() {
   Serial.println("[INFO] Heartbeat detected!");
}

// Web response
void handleRoot() {
Serial.println("[INFO] Web request received. Sending data...");

   String html = "<html><head><meta http-equiv='refresh' content='2'>
<title>ESP32 Pulse Monitor</title></head><body>";
html += "<h1>ESP32 Health Monitor</h1>";
   html += "<p><strong>Heart Rate:</strong> " + String(heartRate, 2) + "
BPM</p>";
   html += "<p><strong>SpO2:</strong> " + String(spO2, 2) + " %</p>";
   html += "<p><strong>Temperature:</strong> " + String(temperatureC, 2) + "
°C</p>";
   html += "<p>Updated every 2 seconds.</p></body></html>";

   server.send(200, "text/html", html);
   Serial.println("[SUCCESS] Data sent to web client.");
} void setup()
{
   Serial.begin(115200);
 Serial.println("\n[START] Initializing ESP32 Health Monitor System...");
   // Buzzer pin setup
 pinMode(BUZZER_PIN, OUTPUT);
digitalWrite(BUZZER_PIN, LOW);
   // OLED setup
 if (!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
      Serial.println("[ERROR] SSD1306 OLED not found!");
      while (true);
   }
```

5.5 SAMPLE OUTPUT



Feel Better About Finding HealthCare

Advanced IoT health monitoring with AIML diagnostics.



Our Services

Real-Time Health Monitoring

Track vital signs including heart rate, blood pressure, body temperature, and environmental humidity through our IoT-connected medical devices.

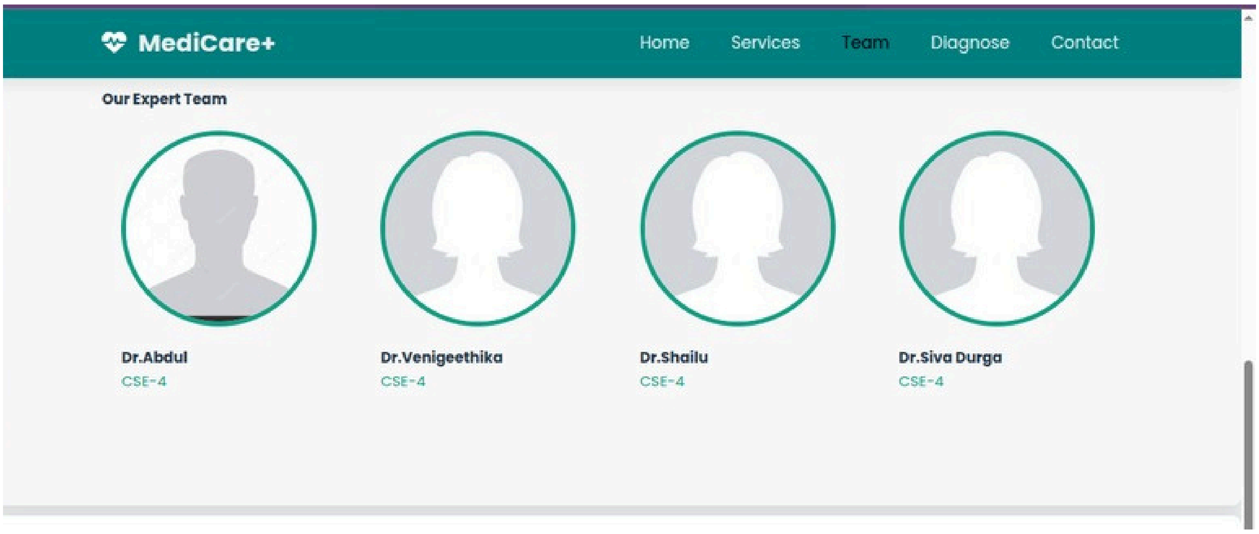
Comprehensive Health Analysis

Our system analyzes your data to provide clear health status indicators (Healthy/Unhealthy) with detailed metrics and trends over time.

Instant Health Alerts

Receive immediate notifications if any parameters fall outside normal ranges, with suggested actions based on severity levels.

Our Expert Team



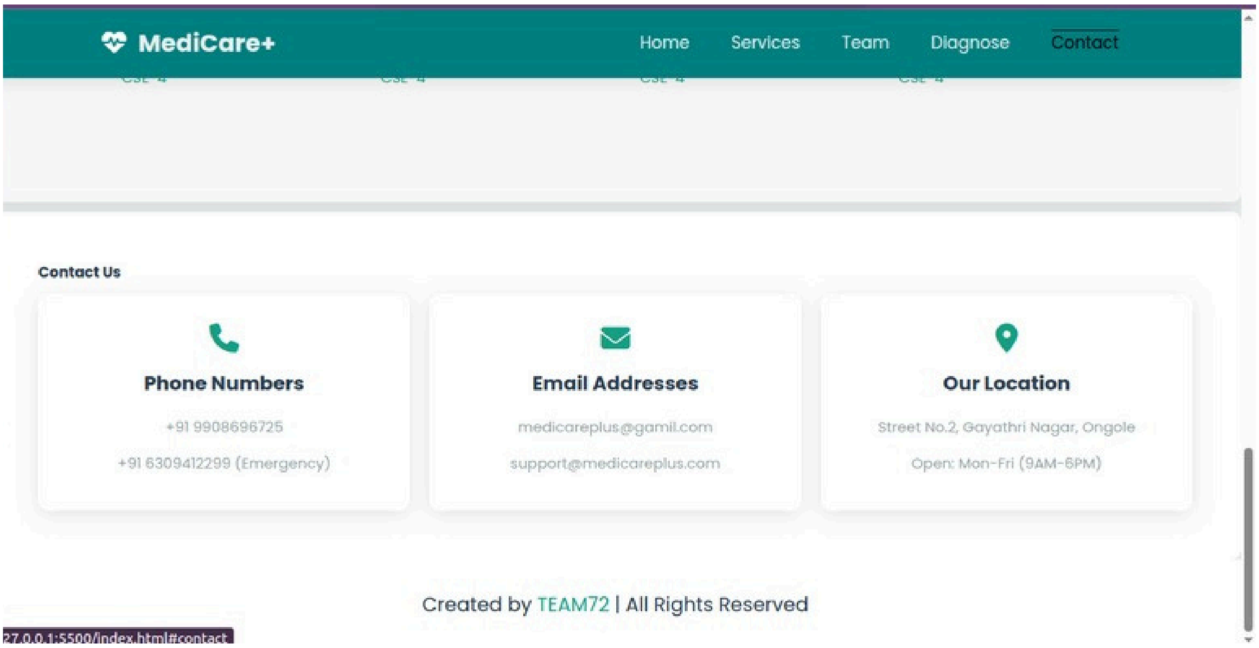
Our Expert Team

Dr.Abdul
CSE-4

Dr.Venigeethika
CSE-4

Dr.Shailu
CSE-4

Dr.Siva Durga
CSE-4



Contact Us

Phone Numbers

+91 9908696725

+91 6309412299 (Emergency)

Email Addresses

medicareplus@gamil.com

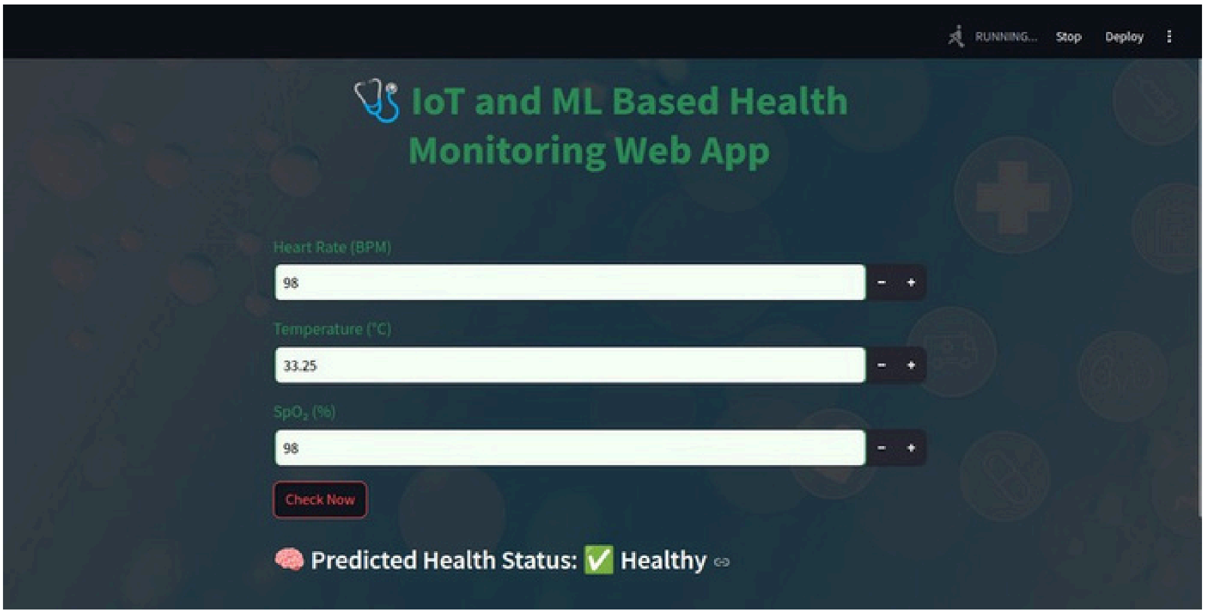
support@medicareplus.com

Our Location

Street No.2, Gayathri Nagar, Ongole

Open: Mon-Fri (9AM-6PM)

Created by TEAM72 | All Rights Reserved



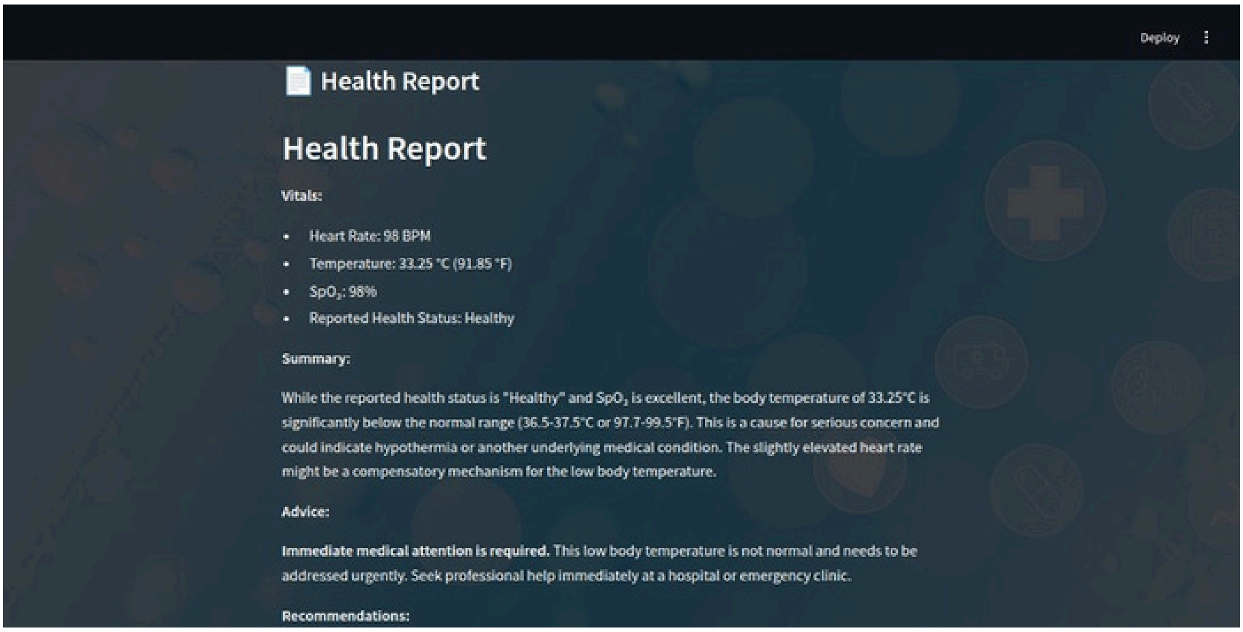
IoT and ML Based Health Monitoring Web App

Heart Rate (BPM): 98

Temperature (°C): 33.25

SpO$_2$ (%): 98

Predicted Health Status: ✅ Healthy



Health Report

Health Report

Vitals:

- Heart Rate: 98 BPM
- Temperature: 33.25 °C (91.85 °F)
- SpO$_2$: 98%
- Reported Health Status: Healthy

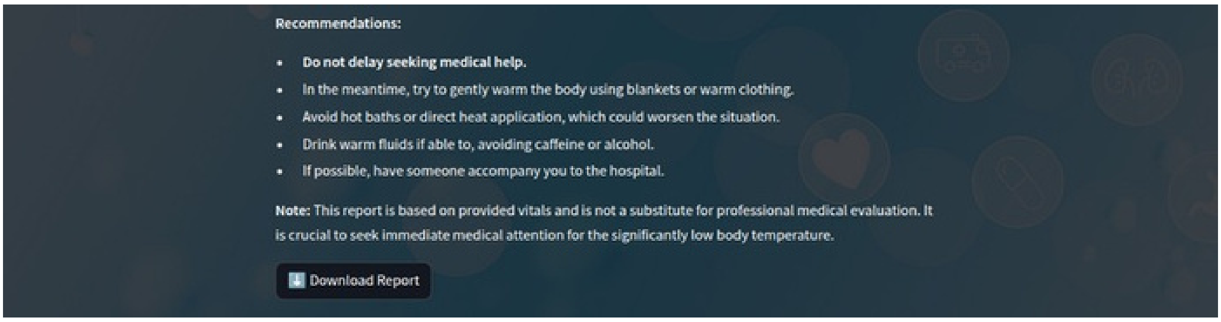
Summary:

While the reported health status is "Healthy" and SpO$_2$ is excellent, the body temperature of 33.25°C is significantly below the normal range (36.5-37.5°C or 97.7-99.5°F). This is a cause for serious concern and could indicate hypothermia or another underlying medical condition. The slightly elevated heart rate might be a compensatory mechanism for the low body temperature.

Advice:

Immediate medical attention is required. This low body temperature is not normal and needs to be addressed urgently. Seek professional help immediately at a hospital or emergency clinic.

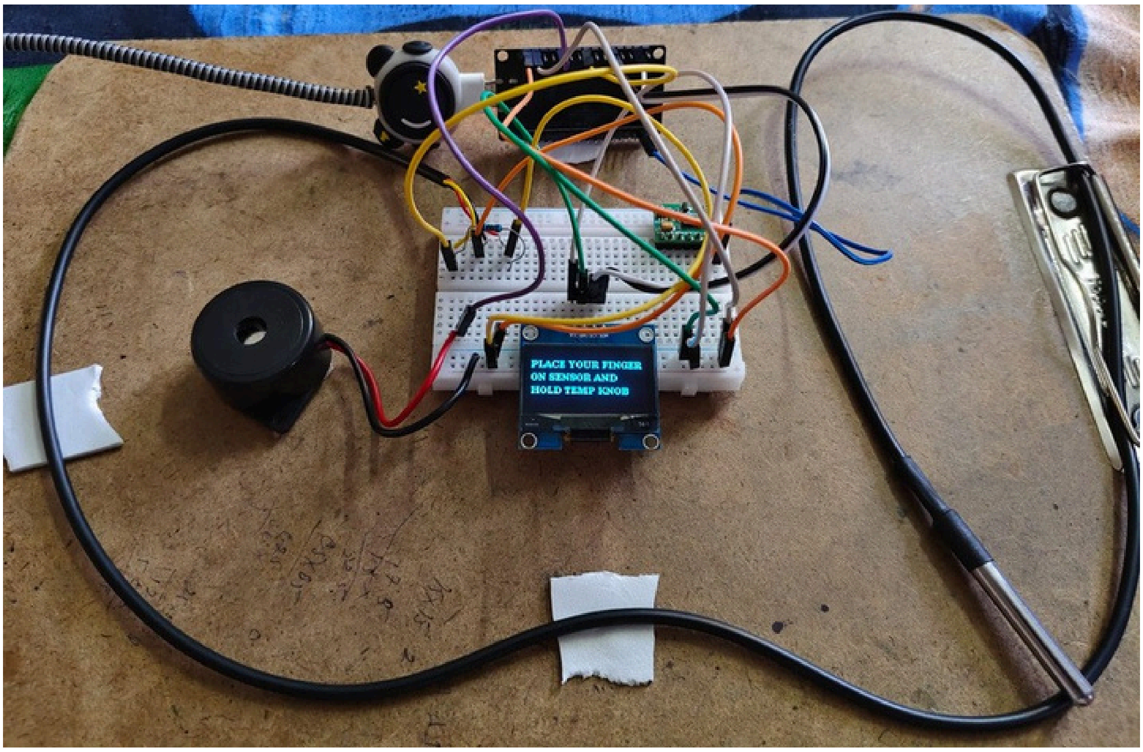
Recommendations:

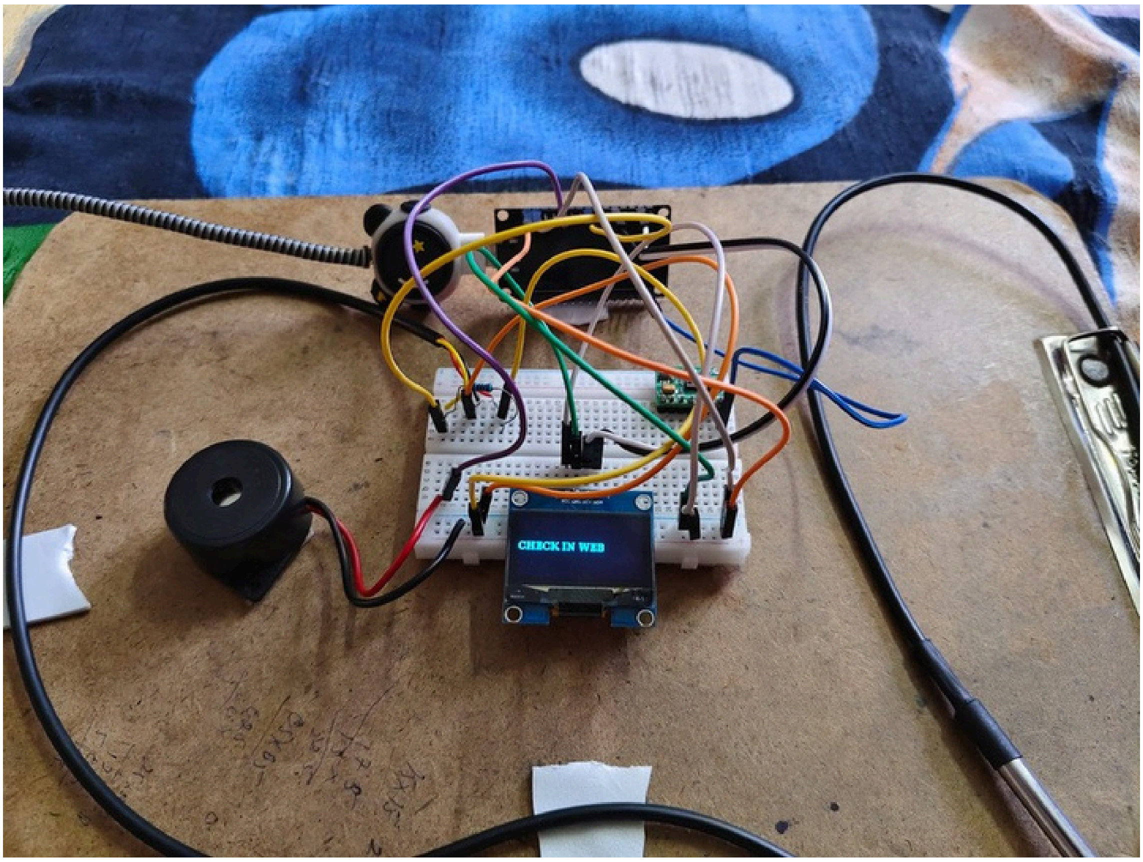


Recommendations:

- **Do not delay seeking medical help.**
- In the meantime, try to gently warm the body using blankets or warm clothing.
- Avoid hot baths or direct heat application, which could worsen the situation.
- Drink warm fluids if able to, avoiding caffeine or alcohol.
- If possible, have someone accompany you to the hospital.

Note: This report is based on provided vitals and is not a substitute for professional medical evaluation. It is crucial to seek immediate medical attention for the significantly low body temperature.


PLACE YOUR FINGER
ON SENSOR AND
HOLD TEMP KNOB


CHECK IN WEB

6. CONCLUSION

This project successfully demonstrates the potential of integrating IoT, machine learning, and generative AI to create an intelligent, real-time health monitoring and diagnostic system. By utilizing the ESP32 microcontroller in conjunction with vital sensors such as the MAX30100 (for heart rate and SpO_2) and a temperature sensor, the system continuously collects and broadcasts real-time health data over a locally hosted web server. This data is seamlessly fetched by a Streamlit-based web application that not only visualizes the vitals but also performs predictive analysis using a trained Random Forest Classifier to determine whether the user's health status is within a healthy range. Going a step further, the system incorporates the power of generative AI through Google's Gemini API to generate personalized, text-based health reports. These reports provide not just a summary of the user’s vitals and diagnosis, but also offer contextual advice and light prescription-like guidance, making the output meaningful and easier to understand for users without a medical background. Additionally, the one-click text file download feature enhances usability and provides a shareable format for consultation or future reference. The project addresses several limitations of traditional healthcare systems, including the lack of continuous monitoring, the absence of immediate alerts, and limited access to professional healthcare in remote or underserved regions. Its modular design and lightweight architecture make it portable and highly scalable, opening doors to further enhancements such as mobile integration, cloud storage, and advanced sensors for expanded diagnostics. In conclusion, this project not only meets its primary objective of building a real-time, AI- powered health monitoring system but also sets a strong foundation for future innovations in remote and personalized healthcare. It holds significant potential to improve healthcare delivery, especially in rural or resource-constrained environments, by enabling timely detection, preventive action, and intelligent decision-making support.

7. FUTURE ENHANCEMENT

1. Cloud Integration for Remote Access
   - Deploy the data and reports to cloud platforms like Firebase, AWS, or Azure for global accessibility and long-term storage.
   - Enable real-time data syncing across devices and healthcare providers.
2. Emergency Alert System
   - Incorporate SMS, email, or app notifications in case vitals exceed safe thresholds.
   - Integrate with APIs like Twilio or SendGrid for timely alerts.
3. Mobile Application Development
   - Build a companion mobile app for on-the-go monitoring and report access.
   - Use cross-platform frameworks like Flutter or React Native for wider reach.
4. Telemedicine Integration
   - Allow patients to consult healthcare professionals directly through the app based on the AI-generated reports.
   - Integrate video calling and chat support with tools like WebRTC.
5. Enhanced Sensor Support
   - Add advanced sensors to monitor ECG, blood pressure, blood glucose, respiration rate, etc.
   - Enable modular sensor expansion to adapt to individual patient needs.
6. Data Logging and Visualization
   - Implement time-series graphs for vitals over days/weeks/months.
   - Provide historical comparisons and trend-based health insights.
7. User Authentication and Profiles
   - Create secure login systems for multiple users.
   - Enable personalized dashboards and medical histories.
8. Voice-Activated AI Assistance
   - Integrate voice interaction using tools like Google Assistant or Alexa for accessibility.
   - Offer spoken health summaries for visually impaired or elderly users.
9. Offline Functionality
   - Allow the system to store data locally and sync when internet connectivity is restored.
   - Ensure uninterrupted monitoring in rural or remote areas.
10. AI-Driven Smart Recommendations
    - Extend the AI module to provide health improvement suggestions, lifestyle tips, or medication reminders based on vitals and trends.

REFERENCES

[1] scikit-learn documentation: https://scikit-learn.org/

[2] Google Generative AI: https://ai.google.dev/

[3] Streamlit Docs: https://docs.streamlit.io/

[4] ChatGPT: https://chatgpt.com/

[5] Kaggle Datasets: https://www.kaggle.com/datasets

[6] Random Forest Classifier: https://www.datacamp.com/tutorial/random-forests-classifier-python

[7] MAX30100 Pulse Sensor Datasheet

[8] ESP32 Wi-Fi and Web Server Examples from Espressif

[9] Research articles on IoT-based healthcare systems

[10] Nabila Sabrin Sworna, A.K.M. Muzahidul Islam, Swakkhar Shatabda, Salekul Islam,Towards development of IoT-ML driven healthcare systems: A survey,Journal of Network and Computer Applications,Volume 196, 2021, 103244, ISSN 1084-8045, https://doi.org/10.1016/j.jnca.2021.103244. https://www.sciencedirect.com/science/article/pii/S1084804521002423

[11] Saif, S., Jana, M., Biswas, S. (2021). Recent Trends in IoT–Based Smart Healthcare Applying ML and DL. In: Tavares, J.M.R.S., Chakrabarti, S., Bhattacharya, A., Ghatak, S. (eds) Emerging Technologies in Data Mining and Information Security. Lecture Notes in Networks and Systems, vol 164. Springer, Singapore. https://doi.org/10.1007/978-981-15-9774-9_72

[12] Role of IoT and ML in Healthcare (K. K. . Vaigandla , Trans.). (2025). Babylonian Journal of Artificial Intelligence, 2025, 23-36. https://doi.org/10.58496/BJAI/2025/003

[13] Venkatesh, Dr.A. Narasima, Reimagining the Future of Healthcare Industry through Internet of Medical Things (IoMT), Artificial Intelligence (AI), Machine Learning (ML), Big Data, Mobile Apps and Advanced Sensors (October 28, 2019). Available at SSRN: https://ssrn.com/abstract=3522960 or http://dx.doi.org/10.2139/ssrn.3522960