Logo, company name

Description automatically generated

Team

PHOENIX

**IoT-Enabled Smart Drug Monitoring System**

**for** **Tracking Patient Treatments**

A person in a suit and tie

Description automatically generated with medium confidence

A picture containing water, sky, outdoor, person

Description automatically generated**A person in a black dress

Description automatically generated with low confidence**

M.A. Isuru Munasinghe

Undergraduate of Electronic

& Telecommunication Department

University of Moratuwa

Avishka Rathnavibushana

Undergraduate of Computer Science Department

University of Moratuwa

V. Yomali Lokugama

Undergraduate of Electronic

& Telecommunication Department

University of Moratuwa

**ABSTRACT**

Nowadays health sector is developing based on wireless sensing node technology. Paying attention for each patient on time is very paramount in any hospital. Also, due to the increase in the number of patients during such a corona epidemic, it is very difficult for the nursing staff to pay special attention to the patients and provide medicines. Therefore, we intend to aid that problem by bringing in an automatic control system which will provide medication on time for every patient, updating his/her medical background and whenever the doctors prescribe medicines or scans for them that will automatically inform relevant medical staff. This device uses IR and ToF sensors for tracking both pills and syrup levels. Other than that condition of the container measured via temperature and humidity sensors and indicate when there is an unnecessary environment for pills and syrups also. LCD display shows patients details and the number of pills and syrups he/she consume for a given time. That will help doctors to track the medical condition of the patient without any doubt whether they took medicines on time or not.

Along with IoT we intend to implement this device with low power consumption. For that we intend to use some low power communication protocols that are not listening to servers in every time. Such as MQTT application layer protocol, IEEE 802.11ah -Wifi protocol. This device collects sensor data and send them to server via ESP 8266 wifi module using Mosquitto MQTT broker with the help of Publish Subscribe architecture.

Google cloud platform is used to deploy frontend and backend applications. For the frontend application, react library is used, and it is deployed as a static application in the cloud. For the backend application Node.js and express.js is used to build the application and deployed in cloud with the help of PM2, production process manger to handle background process. Since MySQL is used as the database, in the cloud we use cloud SQL to host MySQL database in the google cloud.

**TABLE OF CONTENT**

Page No.

Abstract ……………………………………………………………………….…......01

1. Chapter 01 – Introduction
   1. Overview……………………………………………………………………..03
   2. Objectives…….……………………………………………………………....03
   3. Methodology……………………………………………………………....... 03
2. Chapter 02 – Hardware implementation
   1. Introduction…………………………………………………………….…… 06
   2. Hardware components……………………………………………….……… 06
   3. Power……………………………………………………………….……….. 07
   4. Schematics and PCB layouts………………………………………….…….. 08
3. Chapter 03 – Mechanical design
   1. Introduction……………………………………………………………….….10
   2. Computer-Aided Designs………………………………………………….…10
4. Chapter 04 – Software implementation
   1. Introduction……………………………………………………………….….11
   2. Software flow…………………………………………………………….…..11
   3. Protocols……………………………………………………………….……..12
   4. User interfaces…………………………………………………………..……13
5. Chapter 05 – Conclusion…………………………………………………….…...16
6. Chapter 06 – Risks and Challenges………………………………………..….….16
7. Chapter 07 – References…………………………………………………..….…..17

**C****HAP****TER 01 – INTRODUCTION**

**1.1 Overview**

Due to the increasing number of patients in the hospitals, it is difficult for the nursing staff of the hospitals to maintain it properly and efficiently. For this reason, the chance of a single attention on a patient is minimized. Also, the disease of the patients in the ward and the prescriptions prescribed by the doctors are different from each other. As a result, nursing staff may be unable to provide medication on time and paying attention to their medical records. This inefficiency can be caused to make the patient's condition worse. During such a pandemic, it is difficult for the nursing staff to visit and treat the patient and check whether the corona patient received the medication on time frequently. As a solution for that problem, we decided to implement the following solution using IoT.

**1.2 Objectives**

We proposed to create a medicine container with automatic control capability which provide timely medication with alarm, patient’s updated details by doctors and nurses, medication prescribed by the doctor and details of diagnosis reports. This eliminates the need of nursing staff to give different medications from patient to patient on time, as well as to memorize patient medical records and diagnostic data. Also, it has the ability to deliver not only tablets but also syrups in the required quantity at the right time automatically. Once the doctor prescribe medication the system will update current details of the patient. According to that, the nurses fill out the medication and include the time periods for which it should be given to the patients. An alarm will sound when the correct time is reached, and the available medicine in the device will be released. In addition, the nurses and doctors can monitor the dosage of consumed medication.

**1.3 Methodology**

We have created an IoT based system which can solve all the above problems.

![Diagram

Description automatically generated]()

A picture containing kitchen appliance, appliance, coffee maker, food processor

Description automatically generated

*Figure 1.3.1 – Block diagram*

**Use Cases**

* Filling out the patient information form by nurse.

From the nurse UI select add patient tab from navigation bar and fill patient data. After filling out all the fields in the form, submit the form. Then it calls a http post request and calls the backend API for saving patient data in the MySQL database.

* Admit the patient to the ward.

When a patient is being admitted to the ward, the nurse has to go to the patient detail page to enter ward and bed numbers and then press the admit patient button. Then it calls a http post request and calls the backend API to change patient state and publish patient details. After, the IoT device which subscribe the specific topic can get the published details and display in LCD of the monitoring device.

* The Doctor diagnoses the disease and prescribes medicine.

While a doctor diagnoses the disease of the patient, he/she has to add medication details and medical testing to be done, using the add prescription page in the doctor application. After the doctor submitting the form, the data will be sent to the back end and saved in the database.

* Filling out container details by nurse.

Once the doctor creates a prescription for a patient, the nurse can go to the patient details page, prescription tab to view prescription details. From that UI nurse can choose container slots for each medicine, add amount of medication, and add testing progress for the prescription. Then press the update button. After the button is pressed data will be sent to the back end. Then from the back end it first stores the data and publish the data. Then the IoT device which subscribe the specific topic can get the published details. After that, the nurse fill-up medicine according to the prescription into corresponding containers.

* Process of medication usage of the patient.

The nurse should place the medication prescribed by the doctor in the relevant tablet compartment of the IoT device. It has eight compartments that can hold eight different types of tablets and two compartments that can hold two types of syrups at once. After the tablets have been inserted into the correct chambers, it should be covered with two airtight covers to prevent exposure to air. A password must be entered to open the partition where the tablets are to be inserted into the device, thereby preventing anyone other than the hospital staff from modifying the device's medication and data. Also, the tablet and the syrup have two syrup containers and a tablet container attached to the device to drop when they come out separately.

When medication time occurs, the container which contains a maximum of 8 tablet types, rotates and releases the tablet one by one into the path where the tablet should travel. For that, the container number should be kept in mind in the microcontroller memory. Data on how to dispense medicines and relevant container numbers should be subscribed through MQTT as a Json object. More information on MQTT topics can be found in the software implementation section. We have installed a gate to release the tablet and use a gear mechanism to open and close it immediately. In addition, an IR beam is used to determine the number of tablets released from each container slot accurately. The height of the syrup is used to determine the syrup volume to be released and it stops flowing when the required volume is reached. Once all the medication has been dropped in correct proportions, a buzzer, and a blinking emergency light signal alert the patient.

* Environment warning notification.

The quality of drugs depends on the temperature and humidity of the environment. It is not advisable to store medicines especially in high humidity environments. If such an unfavorable environmental situation occurs in the container, relevant notifications will be sent to inform the nurses about it. When a topic related to a warning is published from the IoT device, the backend server grabs the message since it is subscribed to catch warning details. Then that data is saved in the database and notify by the nurse. In the nurse's UI these notifications are visible in the notification page.

* Periodic environment details update.

The temperature and humidity in the medicine container are updated every hour. When a topic related to a timely environment detail is published from the IoT device, the backend server grabs the message since it is subscribed to catch warning details. Then that data is saved in the database. These timely details can be viewed in the patient detail’s environment tab.

**CHAPTER 02 – HARDWARE IMPLEMENTATION**

**2.1 Introduction**

When designing hardware, size of the device, user safety, user-friendly environment, and device cost are the main factors we need to consider. We intend to implement a device that can be mounted near to the patient. Then patient can easily take his medicine without any difficulty. Therefore, the device must ensure the patient’s safety. For that it must have a good electrical system and mechanical system.

**2.2 Hardware components**

Components we used:

* Microcontroller Unit – Arduino Mega 2560 Board
* A NodeMCU development kit with ESP8266 Wi-Fi SoC is used to implement this system.
* VL53L0X Time of Flight sensors used to measure the liquid level.
* DHT22 Humidity and temperature sensor used to measure the environmental condition of the container.
* FUXU NE555 IR emitter and transmitter used to count pills.
* LCD 20x4 display with I2C module, used to display all the necessary information of the patients.
* Buzzer is being used to inform the patients when the medication time occurred.
* SG 5010 servo motors used to achieve the accurate angular or linear position and for specific velocity and acceleration.

We intend to use Arduino Mega 2560 board with ESP8266 Wi-Fi module to control and process all commands, and the reliability of the product can be enhanced by using a Raspberry Pi board. The PWM and Digital pins are used for servos and sensor connections respectively.

**Sensor Panel & Actuators**

* A picture containing text, electronics, circuit

  Description automatically generatedIR receiver & Transmitter

FUXU NE555 IR receiver and transmitter is used to determine the number of pills using an open loop controlling technique. It is important to measure it accurately. This is achieved by placing the IR transmitter and receiver pointing at each other. If a pill falls through the IR beam, it will block the transmitted signal which means that a pill has fallen through it[1].

Figure 2.2.1

FUXU NE555

* A picture containing electronics

  Description automatically generatedHumidity and temperature sensor

The quality of some medicines depends on the humidity and temperature of the environment. Therefore, it is essential to keep the humidity and temperature in the container at a favorable level. DHT22 module is used to measure both humidity and temperature using a capacitive humidity sensor and a thermistor with an accuracy of ±2% and ±0.5ºC respectively[2].

Figure 2.2.2

DHT22 sensor

* A close-up of a microchip

  Description automatically generated with low confidenceToF sensor

The doctor prescribes not only tablets but also syrups to the patient. We have designed this device that can give the patient even two types of syrups. We mounted a VL53L0X ToF sensor on the top surface of the syrup container to accurately measure the amount of syrup required to deliver. It precisely measures the volume to be delivered using the liquid stem height calculation technique. Also, this sensor has 1 mm resolution up to 8cm and it is best suited for IoT platform devices due to its energy saving capabilities[3].

Figure 2.2.3

VL53L0X ToF sensor

* Servo Motors

A picture containing adapter, light

Description automatically generatedRotational actuators have to be used to select the type of tablet to be given by rotating the tablet container, keeping the aperture open until the syrup volume to be released, and delivering the drug in a timely manner. Also, different mechanical mechanisms should be used for the above separate functions, and for this, the servo motors should be rotated to the corresponding angles with different rotational acceleration and velocity. The prototype we propose to implement uses an SG5010 servo motor which has a rotational angle of 180 degrees with fast control response (operating speed of 0.16sec/º60) and stall torque of 6.5kg/cm at 6.6V[4].

Figure 2.2.4

SG5010 Servo motor

**2.3 Power**

We intend to use an AC to DC 12V - 2A power adapter to operate our medical device. Then the 12V is efficiently reduced to 5V and 6V using Pololu D24V22F5 and D24V22F6 voltage regulators to obtain the required 5V for the sensors and 6V for the actuators. These buck converters can deliver typical continuous output currents between 1.8 A and 2.5 A, depending on the input voltage. Especially in the module there is an enable input that can be used to put the regulator in a low-power state with a current draw of 5 µA to 10 µA per Volt. In addition, a 12v Li-Ion battery pack is used to prevent the device from malfunctioning in the case of an emergency power breakdown.

The device is operated supporting two modes to reduce the power consumption which is an important feature of an IoT device.

1. Low power sleeping mode – All the sensors will turn off (sleeping mode) and only leave the microcontroller unit into low power mode. In this case no data is received from the sensors and no data is sent to the actuators.

2. Operating mode – In this mode, the microcontroller usually listens to MQTT requests for controlling the actuators and receives sensor readings from selected sensors.

**2.4 Schematics and PCB layouts**

![Diagram, schematic

Description automatically generated]()PCBs are designed to connect electronic component parts to microcontroller unit. Schematics and PCBs are designed using Altium Designer software. For better connections between components, the paths are routed in such a way that 90-degree angles are removed, and all grounds are common by applying a copper pour to the bottom layer.

![Diagram

Description automatically generated with medium confidence]()

*Figure 2.4.2 – PCB Layout*

*Figure 2.4.1 – Schematic diagram*

![A picture containing text, electronics, circuit

Description automatically generated]()![A screenshot of a computer

Description automatically generated with low confidence]()![Diagram

Description automatically generated]()

Refer the following link to view Schematics and PCBs.

<https://drive.google.com/drive/folders/16vqMbg7GBdRSKYcchr2nor_RpCx1QOPa?usp=sharing>

*Figure 2.4.5 – PCB shield*

*Figure 2.4.4 – PCB bottom layer*

*Figure 2.4.3 – PCB Top layer with copper pour*

**CHAPTER 03 – MECHANICAL DESIGN**

**3.1 Introduction**

Serve motors are used to accurately control all rotations inside the tablet container. Therefore, compound gear system is used to deliver the higher torque needed to rotate the tablet container and to quickly close the aperture. We intended to laser cut some parts inside the container using acrylic.

**3.2 Computer-Aided Designs**

The 2018 version of Solidworks CAD designing software is used to implement the 3D view of the product. Some figures of important parts and assembled final product are given below.

Refer the following link to view Computer-Aided designs.

<https://drive.google.com/drive/folders/1E--n_8XRcptUh8vxLN5RZZ4pGAMpCnBu?usp=sharing>

Graphical user interface

Description automatically generated with medium confidenceDiagram

Description automatically generated

*Figure 3.2.1 – Fully assembled final product*

*Figure 3.2.2 – Assembled parts of syrup container*

*Figure 3.2.3 – Assembled parts of the tablet container*

A picture containing clock

Description automatically generatedA picture containing shape

Description automatically generatedA picture containing diagram

Description automatically generated

Chart, funnel chart

Description automatically generated

**CHAPTER 04 – SOFTWARE IMPLEMENTATION**

**4.1 Introduction**

To implement the system from IoT side we intend to use Arduino software. For better implementation Raspberry pi software also can be used. Eclipse mosquito open-source message broker is used to establish the Publish Subscribe architecture. From user side, frontend application of the system is developed using React library[5] and backend application is developed using Node.js[6] and express.js. As the database, MySQL is used to store data in the system. All three systems frontend, backend and database is deployed and hosted in Google cloud platform. Since backend needs to be run on the cloud while listening to the API calls and subscribed MQTT broker, PM2 is used as the background process manger in the GCP. For the front end, application is built and deployed in the GCP as a static application. To deploy the MySQL data base in the GCP, Cloud SQL is used as service that helps to manage, setup and maintain database.

**4.2 Software flow**

* Basically, we hope to use Arduino in-built libraries, ESP8266 WiFi libraries and PubSubClient MQTT client library. In addition, ESPAsyncTCP[7] and ESPAsyncWebServer[8] libraries have to be used. ESPAsyncTCP is an asynchronous TCP library which allows multi connection network environment for ESP8266 based systems. ESPAsyncWebServer is a library based on ESPAsyncTCP and it can create an asynchronous server instance running on ESP8266. In this application scenario, the system needs to handle multiple concurrent connections (ex: dealing with user inputs and outputs), therefore it is important to use an asynchronous approach which can easily integrate with the user interface. On ESP8266, an asynchronous server listening to http traffic is created instead of a normal WiFi server.
* Using PubSubClient library, an MQTT client instance is created on ESP82266 which is connected to a MQTT broker. (“test.mosquitto.org”[9]).
* ESP8266 is connected to the WiFi LAN in station mode. MQTT client connection drives on top of this.
* Any user input filled by the user under any field will be collected and and published to the MQTT broker under relevant topics.
* Real time information that are stored in cloud will be published to MQTT server and they are subscribed by the nodeMCU. On the other hand, data coming from sensors are processed by the microcontroller and publish those into relevant topics to MQTT via nodeMCU. And they are subscribed by the backend server.
* The relevant data in the cloud are published to following topics (these same topics are subscribed by nodeMCU via MQTT). Those data are published and subscribed on topics unique to the bed ID that will ensure secure data transmission through the MQTT broker.
* wardNumber/bedID/patientDetails : patients details (Name, Age, Gender, description given by the doctor in brief)
* wardNumber/bedID/tablet/medication: Json object contains a list where each item includes container slot ID (1,2,…,8), tablet, timeType(1- time[Morning/Evening/Night], 2-repititive time period) and time.

Ex : {{container :1, tablet: “Panadol”, timeType:1, time: “Morning/Night” },…}

* wardNumber/bedID/syrup/medication : Json object contains a list where each item includes container slot ID (1,2), Syrup, timeType(1- time[Morning/Evening/Night], 2-repititive time period) and time.

Ex : {{container :1, Syrup: “Cough\_Syrup”, timeType:1, time: “Morning/Night” },…}

* The relevant data in the microcontroller are published to following topics through ESP8266 (these same topics are subscribed by backend Server via MQTT)
* wardNumber/bedID/containerID/consumedTablets : Number of tablets consumed by the patient in given tablet container slot.
* wardNumber/bedID/containerID/consumedSyrup : Volume of syrups consumed by the patient in given syrup container slot.
* wardNumber/bedID/containerEnvironmentWarning: Notification if the container’s environment (humidity / temperature) become abnormal.
* wardNumber/bedID/containerEnvironment : Json object that contains humidity and temperature values for every 1 hour.

Ex : {{humidity :20, temperature:25}}

* Servo.h, DHT.h and Adafruit\_VL53L0X libraries are used to drive servo motors and corresponding sensors.
* In user application, each role has specific user interfaces to input data. After those data will be sent to back end using http post calls. Then according to the http call a specific function is called according to the API implemented in the backed. After that data will be stored in the MySQL database. For use cases where data needs to be sent to the IoT device, those data data will be sent to MQTT broker using MQTT protocol with mqtt.js[10] library in node.js.
* When a topic is published from the MQTT broker, they will be grabbed by the backend server and store data in the database. So, whenever user needs to see those data, they can be acquired by the user using relevant http calls.
* Reliability and QoS – Since IoT is dealing in constrained environment there are so many things we need to consider. Such as data transmission, handling data losses etc. Higher QoS level means higher reliability. Here we need to have reliable data transfer as we are dealing with human lives. Such as MQTT with level 2 QoS. Since MQTT uses TCP protocol in transport layer it also ensures the reliability of communication.

**4.3 Protocols**

* HTTP: This application layer protocol is used to transfer data between frontend and backend which is hosted in cloud based on client server architecture.
* MQTT: The entire system is implemented in a publish-subscribe architecture via an MQTT broker Other than these application layer protocols, in lower layers normal internet protocol suite is used (TCP/IP). Publishers and subscribers don’t need to know each other. Therefore, events can handle asynchronous way allowing greater scalability and flexibility. Both publishers and subscribers rely on MQTT broker then infrastructure can be easily scaled up.
* IEEE 802.11ah (Wifi HALOW): Some of the key features of this protocol will help to implement our system in an efficient manner. Such as low throughput, high node density, low power consumption and long range. Therefore, we intend to use this protocol for our sensor networks without using conventional WiFi.

**4.4 User Interfaces**

Graphical user interface, application

Description automatically generatedGraphical user interface, application, website

Description automatically generatedGraphical user interface, application

Description automatically generatedGraphical user interface, application

Description automatically generatedGraphical user interface, application

Description automatically generated

*Figure 4.4.2 – Registration/Login screens of doctor & nurse*

*Figure 4.4.1– Application Welcome Screen*

A picture containing text, electronics, screenshot, several

Description automatically generated Graphical user interface

Description automatically generatedGraphical user interface, application

Description automatically generatedGraphical user interface, application

Description automatically generatedGraphical user interface, application

Description automatically generated

Graphical user interface

Description automatically generatedGraphical user interface, application

Description automatically generated

Graphical user interface

Description automatically generatedGraphical user interface

Description automatically generatedGraphical user interface, application

Description automatically generated

*Figure 4.4.3 – User Interfaces of nurse*

Graphical user interface, text, application

Description automatically generated

*Figure 4.4.3 – User Interfaces of doctor*

Refer the following link to view user interfaces designed by Figma online designing software.

<https://www.figma.com/file/hsKABGIFlNuMIsQKhtm8Jv/IOT-project?node-id=82%3A2117>

<https://drive.google.com/drive/folders/1BHRZrLSAi9qPZn8L0R8WNAzJiwt2uKr4?usp=sharing>

**CHAPTER 05 – CONCLUSION**

The internet of things is now considered as one of the feasible solutions for remote tracking systems. The purpose of this project was to implement a device that will supply medication on time for each patient automatically and that will update his/her medicine list in real time after the doctors prescribe medicine for them. Nurses can fill out medicine into the container according to the prescription as shown on user interface for each patient. Here we can conclude that this will be a huge impact in health sector as we implement this system using IoT. Because we don’t need any physical wiring system to connect devices as most of the connections are done through in wireless media.

**CHAPTER 06 – RISKS AND CHALLENGES**

The confidential handling of patient information, accuracy of the system, smooth flow of each device, safety are the main key factors that we need to pay more attention as we are dealing with human lives. And, with MQTT protocol, topic IDs should be unique for a particular patient. Embedded security and privacy preserving mechanisms should be there. That would be a challenge as we need to attach this system for any kind of medical center. We must have a reliable and high QoS level for that. In addition, we need some special infrastructures when laying this system on existing infrastructures. We must use shared infrastructure, shared data, shared network protocols in an efficient manner when implementing the system within this power constrained environment. As IoT is horizontal layered architecture device heterogeneity is there. That would be a challenge for us as we need to connect so many devices in star topology with self-organizing capabilities.

Since database in the cloud contains confidential details of patients, we had to confirm that only authorized members access the platform. And also, this system requires high accuracy and user should have some technical knowledge to use the system. Therefore, after deploying the system in hospital environment, some awareness sessions need to be conducted to show how to use the system.

**CHAPTER 07 – REFERENCES**

[1]*Ti.com*, 2021. [Online]. Available: https://www.ti.com/lit/ds/symlink/ne555.pdf. [Accessed: 08- Oct- 2021]

[2]"DHT22 Datasheet(PDF) - List of Unclassifed Manufacturers", *Alldatasheet.com*, 2021. [Online]. Available: https://www.alldatasheet.com/datasheet-pdf/pdf/1132459/ETC2/DHT22.html. [Accessed: 08- Oct- 2021]

[3]"VL53L0X Datasheet, PDF - Alldatasheet", *Alldatasheet.com*, 2021. [Online]. Available: https://www.alldatasheet.com/view.jsp?Searchword=VL53L0X&sField=4. [Accessed: 08- Oct- 2021]

[4]"SG-5010 datasheet", *Datasheetspdf.com*, 2021. [Online]. Available: https://datasheetspdf.com/pdf-file/633270/TowerPro/SG-5010/1. [Accessed: 08- Oct- 2021]

[5]"React – A JavaScript library for building user interfaces", *Reactjs.org*, 2021. [Online]. Available: https://reactjs.org/. [Accessed: 07- Oct- 2021]

[6]"Node.js", *Node.js*, 2021. [Online]. Available: https://nodejs.org/en/. [Accessed: 07- Oct- 2021]

[7]"GitHub - me-no-dev/ESPAsyncTCP: Async TCP Library for ESP8266", *GitHub*, 2021. [Online]. Available: https://github.com/me-no-dev/ESPAsyncTCP. [Accessed: 09- Oct- 2021]

[8]"GitHub - me-no-dev/ESPAsyncWebServer: Async Web Server for ESP8266 and ESP32", *GitHub*, 2021. [Online]. Available: https://github.com/me-no-dev/ESPAsyncWebServer. [Accessed: 09- Oct- 2021]

[9]"Eclipse Mosquitto", *Eclipse Mosquitto*, 2021. [Online]. Available: https://mosquitto.org/. [Accessed: 09- Oct- 2021]

[10]"mqtt", npm, 2021. [Online]. Available: https://www.npmjs.com/package/mqtt. [Accessed: 09- Oct- 2021]