PHARMACEUTICAL STORAGE MONITORING SYSTEM

MINI PROJECT REPORT

Submitted by

HARESH M (210701066)

HAYAGREEVAN V (210701080)

In partial fulfillment for the award of the degree

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

RAJALAKSHMI ENGINEERING COLLEGE

ANNA UNIVERSITY: CHENNAI 600 025

MAY 2024

BONAFIDE CERTIFICATE

Certified that this project **"PHARMACEUTICAL STORAGE MONITORING SYSTEM**" is the bonafide work of "**HARESH M (210701066) and HAYAGREEVAN V (210701080)"** who carried out the project work under my supervision.

SIGNATURE
Dr.N.Duraimurugan, M.E., Ph.D.

Associate Professor,

Computer Science & Engineering

Rajalakshmi Engineering College
(Autonomous)

Thandalam, Chennai -602105.

Submitted for the **ANNA UNIVERSITY** practical examination Mini-Project work viva voice held on_____

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

We express our sincere thanks to our beloved and honorable chairman MR. S. MEGANATHAN and the chairperson DR. M. THANGAM MEGANATHAN for their timely support and encouragement.

We are greatly indebted to our respected and honorable principal Dr. S.N. MURUGESAN for his able support and guidance. No words of gratitude will suffice for the unquestioning support extended to us by our head of the department Dr. P. KUMAR, M.E., Ph.D. for being ever supporting force during our project work.

We also extend our sincere and hearty thanks to our internal guide Dr.N.Duraimurugan, M.E., Ph.D. for his valuable guidance and motivation during the completion of this project. Our sincere thanks to our family members, friends and other staff members of information technology.

HARESH M (210701066),
HAYAGREEVAN V (210701080)

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATION

ABBREVIATION	ACRONYM
IOT	Internet of Things
HTTP	HyperText Transfer Protocol
TEMP	Temperature
DHT	Digital Humidity and Temperature
SQL	Structured Query Language

ABSTRACT

This project explores the significance of effective storage monitoring in the pharmaceutical industry and investigates the transformative potential of IoT technology in addressing storage-related challenges. The integration of IoT technology into storage monitoring systems enables real-time data collection and intervention, facilitating improved product quality assurance, enhanced supply chain visibility, and streamlined regulatory compliance. This project encompasses the warehouses / dispensaries which monitor the environment conditions like temperature, humidity and light intensity using IoT devices kept at specified locations. This smart device orchestra is controlled by a microcontroller which sends out required data to the management system. When there is a deviation from normal conditions, the system is expected to notify the change so that the data is interpreted by the concerned authorities to enable them to take quick actions to preserve the medicines. It provides effectiveness in maintaining optimal storage conditions and preventing product spoilage. Further research and collaboration can maximize the benefits of IoT-enabled storage monitoring systems in the pharmaceutical industry.

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

In the dynamic landscape of the pharmaceutical industry, ensuring the integrity and efficacy of pharmaceutical products throughout their lifecycle is paramount. Deviations in factors like temperature, humidity, and light exposure can significantly impact product quality and safety. The integration of advanced technologies becomes imperative to meet the rigorous standards and regulatory requirements governing pharmaceutical storage.

The advent of Internet of Things (IoT) technology has heralded a new era in storage monitoring systems, offering real-time insights and actionable intelligence to pharmaceutical stakeholders. By leveraging IoT-enabled sensors and connectivity, stakeholders can proactively monitor storage conditions, detect deviations, and initiate timely interventions to mitigate risks. This enhances product quality assurance and regulatory compliance.

Through a comprehensive examination of existing literature, industry standards, and case studies, we elucidate the critical role of IoT-enabled storage monitoring systems in maintaining optimal storage conditions and safeguarding product integrity.

1.2 SCOPE OF THE WORK

This project can benefit the pharmaceutical industry. For the Pharmaceutical Industry, it will collect real-time data, analyze it for deviations, and trigger alerts if conditions exceed acceptable thresholds. Key components include IOT sensor selection, software development for data analysis, providing real-time dashboard and alerting the management when there is an abnormality.

1.3 PROBLEM STATEMENT

Limited research has been conducted on the causes and economic impacts of expired medicine, yet medicine costs comprise a significant proportion of healthcare expenditure.The lack and wastage of essential medicine is still one of the most serious public health issues globally. The proposed solution is a system that monitors temperature, pressure, humidity and exposure to other gas compositions in medical warehouses or dispensaries and intimates the unfavorable condition of the environment and expiration of pharmaceutical products accordingly.

1.4 AIM AND OBJECTIVES OF THE PROJECT

This project aims to create a pharmaceutical storage monitoring system with a real-time dashboard web application. Objectives are developing hardware and software for data collection, analysis, monitoring and alerting, as well as designing a user-friendly web interface. Additionally, ensuring compliance with regulations and notifying the administrators when there is a change in environment.

CHAPTER 2

SYSTEM SPECIFICATIONS

2.1 IOT DEVICES

1. ESP8266-12E NODEMCU with Wi-Fi Module
2. DHT11 Sensor
3. BH1750 Sensor

2.2 SYSTEM HARDWARE SPECIFICATIONS

PROCESSOR	Intel i5 11th Gen
MEMORY SIZE	8 GB (Minimum)
HDD	40 GB (Minimum)

2.3 SOFTWARE SPECIFICATIONS

Operating System	Windows 11
Front – End	React JS
Back – End	PostgreSQL, Express and Node JS
Browser	Google Chrome
IDE	Visual Studio Code

CHAPTER 3
SYSTEM DESIGN

3.1 ARCHITECTURE DIAGRAM

An architecture diagram is a graphical representation of a set of concepts, that are part of an architecture, including their principles, elements and components

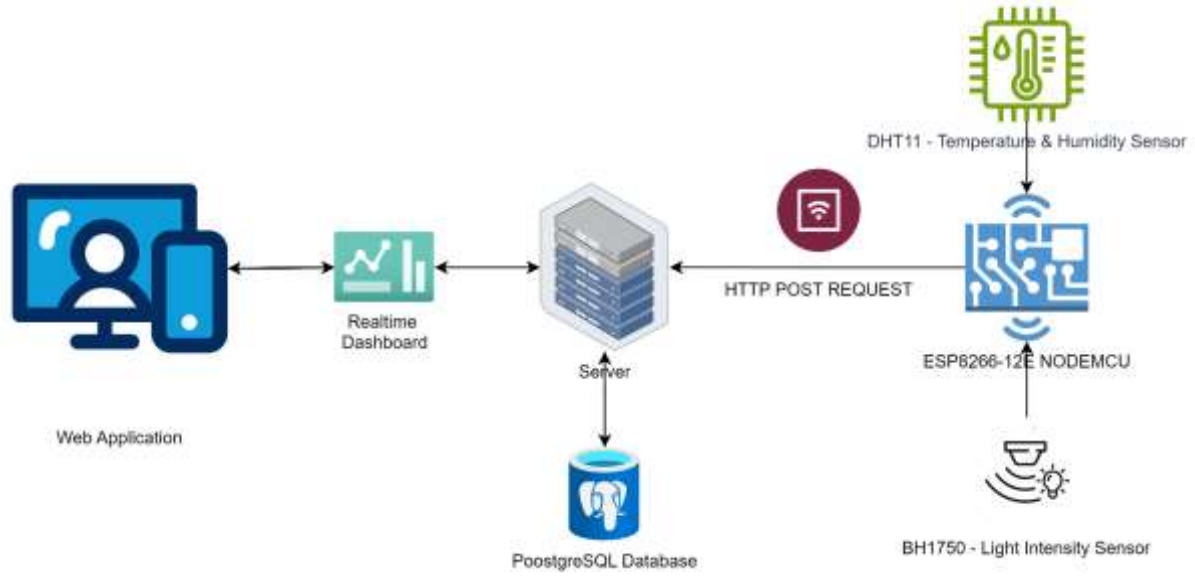

Figure 3.1 Architecture Diagram

From the above Figure 3.1, the architecture of the system is well understood.

3.2 USE CASE DIAGRAM

A use case is a list of actions or event steps typically defining the interactions between a role (known in the Unified Modelling Language as an actor) and a system to achieve a goal. The actor can be a human or other external system.

Figure 3.2 Use case diagram

From the above figure 3.2, the interactions between a role in the system is shown

3.3 ACTIVITY DIAGRAM

An activity in Unified Modelling Language (UML) is a major task that must take place in order to fulfill an operation contract. Activities can be represented inactivity diagrams. An activity can represent: The invocation of an operation. A step in a business process.

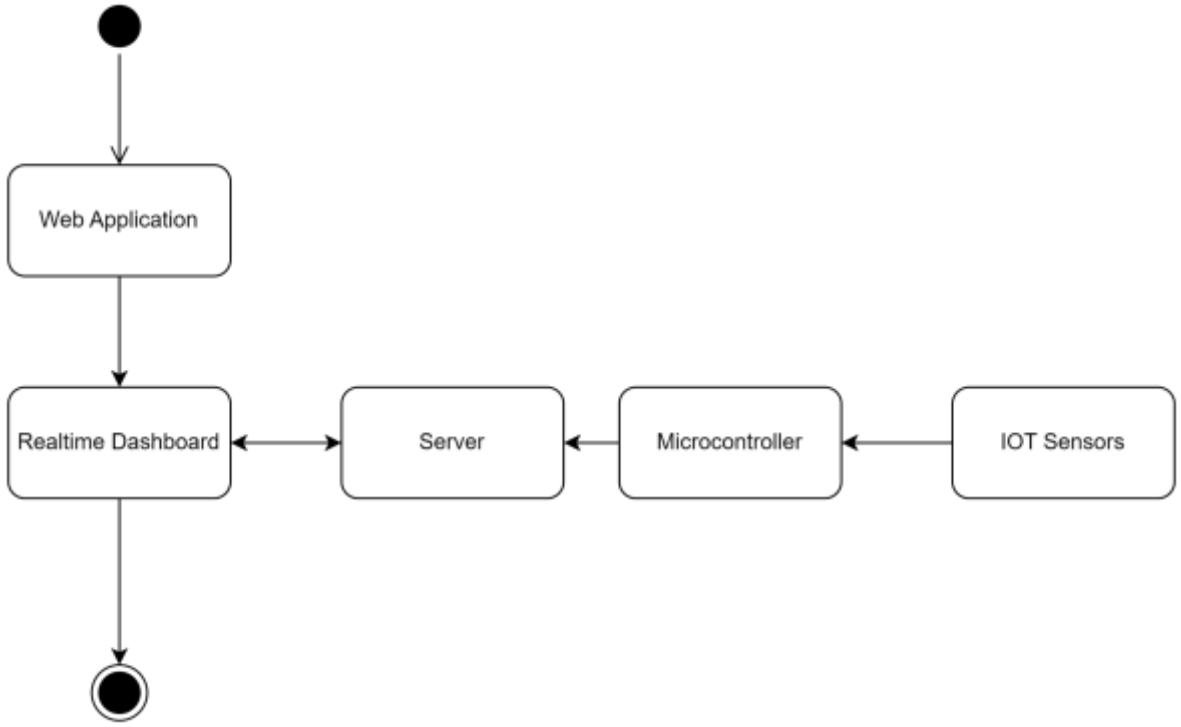

Figure 3.3 Activity Diagram

From the above figure 3.3, the activities of the system are shown

3.4 CLASS DIAGRAM

A class diagram is an illustration of the relationships and source code dependencies among classes in the Unified Modelling Language (UML). In this context, a class defines the methods and variables in an object, which is a specific entity in a program or the unit of code representing that entity.

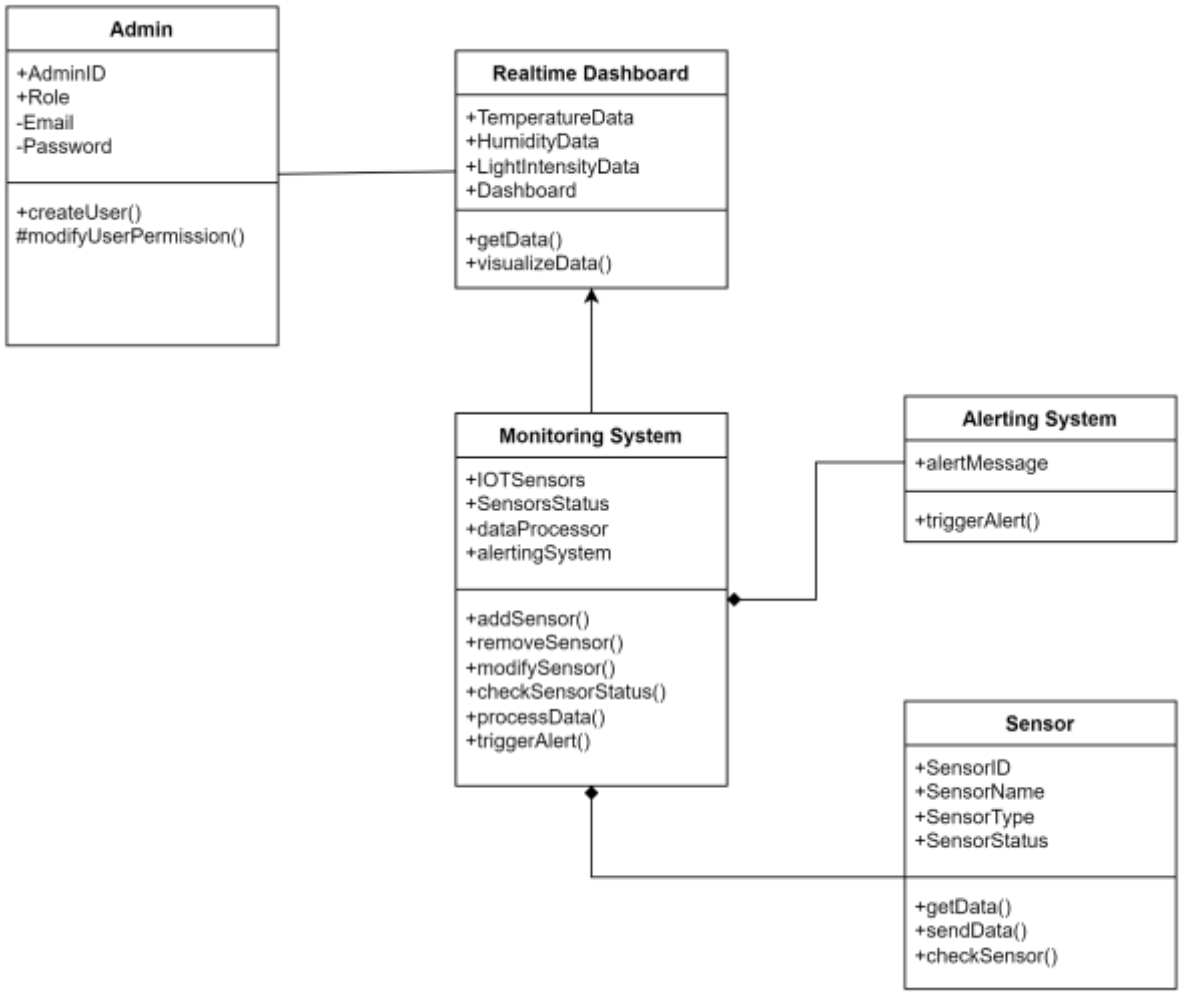

Figure 3.4 Class Diagram

The above Figure 3.4 is the class diagram for the system.

CHAPTER 4

MODULE DESCRIPTION

4.1 HARDWARE MODULE:

This module comprises temperature, humidity, and light intensity sensors along with the microcontroller (ESP8266-12E) for data acquisition. The sensors are responsible for continuously monitoring the environmental conditions in the pharmaceutical storage facilities.

4.2 DATA COLLECTION AND PROCESSING MODULE:

Responsible for interfacing with the hardware module to collect real-time data from the sensors and storing the data into the database. Utilizes http communication protocols to ensure accurate and reliable data transmission. Implements algorithms to analyze the data and identify deviations from predefined thresholds.

4.3 ALERTING MODULE:

Monitors processed sensor data for any deviations from acceptable ranges. Triggers alerts (e.g., email, push notifications) to notify administrator when deviations are detected, ensuring timely intervention to prevent damage to pharmaceutical products.

4.4 WEB APPLICATION MODULE:

Develops a user-friendly web interface with a real-time dashboard for visualizing sensor data. Provides stakeholders with access to real-time monitoring of storage conditions, enabling remote management and decision-making.

4.5 INTEGRATION MODULE:

Integrates the data processing and alerting functionalities with the web application to ensure seamless communication and data display. Enables synchronization between the monitoring system and the dashboard for immediate updates.

CHAPTER 5

TABLE

5.1 MEDICINE TABLE

S.NO	ATTRIBUTE	TYPE
1	MED_ID	NUMBER(5)
2	MED_NAME	VARCHAR(45)
3	MANUFACTURER	VARCHAR(45)
4	PREF_MIN_TEMP	NUMBER(5,2)
5	PREF_MIN_HUM	NUMBER(5,2)
6	PREF_MIN_LIGHT	NUMBER(7,2)
7	PREF_MAX_TEMP	NUMBER(5,2)
8	PREF_MAX_HUM	NUMBER(5,2)
9	PREF_MAX_LIGHT	NUMBER(7,2)

5.2 STORAGE TABLE

S.NO	ATTRIBUTE	TYPE
1.	STORAGE_ID	NUMBER(5)
2.	MED_ID	NUMBER(5)
3.	MFD	DATE
4.	EXPIRY_DATE	DATE

5.3 HISTORY TABLE

S.NO	ATTRIBUTE	TYPE
1.	HISTORY_ID	NUMBER(5)
2.	STORAGE_ID	INTEGER
3.	TIME	TIMESTAMP WITHOUT TIMEZONE
4.	TEMP	NUMBER(5,2)
5.	LUX	NUMBER(7,2)
6.	HUMIDITY	NUMBER(5,2)

5.4 CURRENT DATA TABLE

S.NO	ATTRIBUTE	TYPE
1.	TEMP_IN_C	NUMBER(5,2)
2.	TEMP_IN_F	NUMBER(5,2)
3.	LIGHT	NUMBER(7,2)
4.	HUMIDITY	NUMBER(5,2)

CHAPTER 6

SAMPLE CODING

ESP8266-12E NODEMCU Program

```
#include <ESP8266WiFi.h>
#include <ESP8266HTTPClient.h>
#include "DHT.h"
#include <BH1750.h>
#include <Wire.h>


#define DHTPIN 2
#define DHTTYPE 11


const char* ssid = "FTTH_Hex";
const char* password = "03031204";
String deviceID = "Hex";

DHT dht(DHTPIN, DHTTYPE);
BH1750 lightMeter;


void setup() {
  Serial.begin(9600);
  Serial.println();

  dht.begin();

  Wire.begin();

  lightMeter.begin();

  Serial.println(F("BH1750 Test begin"));
  Serial.print("Connecting to ");
```

```
  Serial.println(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());

}

void loop() {

  delay(2000);

  float h = dht.readHumidity();
  float t = dht.readTemperature();
  float f = dht.readTemperature(true);

  if (isnan(h) || isnan(t) || isnan(f)) {
    Serial.println(F("Failed to read from DHT sensor!"));
  }

  float hif = dht.computeHeatIndex(f, h);
  float hic = dht.computeHeatIndex(t, h, false);

  Serial.print(F("Humidity: "));
  Serial.print(h);
  Serial.print(F("%  Temperature: "));
  Serial.print(t);
  Serial.print(F("°C "));
  Serial.print(f);
  Serial.print(F("°F  Heat index: "));
  Serial.print(hic);
  Serial.print(F("°C "));
```

```
  Serial.print(hif);
  Serial.println(F("°F"));

  float lux = lightMeter.readLightLevel();
  Serial.print("Light: ");
  Serial.print(lux);
  Serial.println(" lx");
  delay(1000);

  HTTPClient http;
  String URL = "http://192.168.1.6:3001";
  WiFiClient client;
  http.begin(client, URL);

  http.addHeader("Content-Type", "application/x-www-form-urlencoded");

  String postData =
"StorageID=1&TemperatureInC="+String(t)+"&TemperatureInF="+String(f)+"&Humidi
ty="+String(h)+"&Light="+String(lux);
  int httpCode = http.POST(postData);
  Serial.println(httpCode);
  if (httpCode > 0) {
    String payload = http.getString();
    Serial.println(payload); // Print response
  }
  http.end();
  Serial.println("END");
  delay(2000);
}
```

Web Application

1. index.js

```
        const express = require("express")
const app = express()
const PORT = 3001
const cors = require('cors');
const morgan = require('morgan');
const bodyParser = require("body-parser");

const corsOptions = require('./config/corsOptions')
const db = require('./config/DBConnect')
const medStorageRoute = require('./routes/med_storage.route');
const historyRoute = require('./routes/history.route');

app.use(bodyParser.urlencoded({ extended: false }));
app.use(cors(corsOptions));
app.use(express.json());
app.use(morgan('tiny'));

db.connect().then(()=>{
   console.log("Postgres is connected");
})

app.get("/getSensorData", async(req, res) => {
   res.send(response)
})

let StorageID, TemperatureInC, TemperatureInF, Humidity, Light;
let currentData;

app.post('/',async(req,res)=>{
   console.log(req.body);
   currentData = req.body;
   StorageID = req.body.StorageID;
   TemperatureInC = req.body.TemperatureInC;
   TemperatureInF = req.body.TemperatureInF;
```

```
    Humidity = req.body.Humidity;
    Light = req.body.Light;

    if(TemperatureInC !='nan' && Light>0){
        await db.query("INSERT INTO HISTORY(STORAGE_ID, TEMP, LIGHT,
HUMIDITY) VALUES($1,$2,$3,$4)",[StorageID,TemperatureInC,Light,Humidity]);
        await db.query("UPDATE CURRENT_DATA SET TEMP_IN_C = $1, TEMP_IN_F =
$2, LIGHT = $3, HUMIDITY = $4",[TemperatureInC,TemperatureInF,Light,Humidity]);
        console.log("Inserted with Light");
    }else if(TemperatureInC !='nan'){
        await db.query("INSERT INTO HISTORY(STORAGE_ID, TEMP, HUMIDITY)
VALUES($1,$2,$3)",[StorageID,TemperatureInC,Humidity]);
        await db.query("UPDATE CURRENT_DATA SET TEMP_IN_C = $1, TEMP_IN_F =
$2, HUMIDITY = $3",[TemperatureInC,TemperatureInF,Humidity]);
        console.log("Inserted without Light");
    }
    res.status(200).send("POST Request Received");
})

app.get('/currentData',async(req,res)=>{
    const result =await db.query("Select * from current_data");
    res.status(200).send(result.rows[0]);
});



app.use('/meds-storage',medStorageRoute);
app.use('/history',historyRoute);

app.listen(PORT, () => {
    console.log("Server is online")
})
```

1. Dashboard.jsx

```
import * as React from 'react';
import { styled, createTheme, ThemeProvider } from '@mui/material/styles';
import CssBaseline from '@mui/material/CssBaseline';
import MuiDrawer from '@mui/material/Drawer';
import Box from '@mui/material/Box';
import MuiAppBar from '@mui/material/AppBar';
import Toolbar from '@mui/material/Toolbar';
import List from '@mui/material/List';
import Typography from '@mui/material/Typography';
import Divider from '@mui/material/Divider';
import IconButton from '@mui/material/IconButton';
import Badge from '@mui/material/Badge';
import Container from '@mui/material/Container';
import Grid from '@mui/material/Grid';
import Paper from '@mui/material/Paper';
import Link from '@mui/material/Link';
import MenuIcon from '@mui/icons-material/Menu';
import ChevronLeftIcon from '@mui/icons-material/ChevronLeft';
import NotificationsIcon from '@mui/icons-material/Notifications';
import { mainListItems, secondaryListItems } from './listItems';
import Chart from './Chart';
import Deposits from './Deposits';
import Orders from './Orders';
```

```
function Copyright(props) {
  return (
    <Typography variant="body2" color="text.secondary" align="center" {...props}>
      {'Copyright © '}
      <Link color="inherit" href="https://mui.com/">
        Your Website
      </Link>{' '}
      {new Date().getFullYear()}
      {'.'}
    </Typography>
  );
}

const drawerWidth = 240;

const AppBar = styled(MuiAppBar, {
  shouldForwardProp: (prop) => prop !== 'open',
})(({ theme, open }) => ({
  zIndex: theme.zIndex.drawer + 1,
  transition: theme.transitions.create(['width', 'margin'], {
    easing: theme.transitions.easing.sharp,
    duration: theme.transitions.duration.leavingScreen,
  }),
  ...(open && {
    marginLeft: drawerWidth,
    width: `calc(100% - ${drawerWidth}px)`,
    transition: theme.transitions.create(['width', 'margin'], {
```

```
      easing: theme.transitions.easing.sharp,
      duration: theme.transitions.duration.enteringScreen,
    }),
  }),
}));

const Drawer = styled(MuiDrawer, { shouldForwardProp: (prop) => prop !== 'open' })(
  ({ theme, open }) => ({
    '& .MuiDrawer-paper': {
      position: 'relative',
      whiteSpace: 'nowrap',
      width: drawerWidth,
      transition: theme.transitions.create('width', {
        easing: theme.transitions.easing.sharp,
        duration: theme.transitions.duration.enteringScreen,
      }),
      boxSizing: 'border-box',
      ...(!open && {
        overflowX: 'hidden',
        transition: theme.transitions.create('width', {
          easing: theme.transitions.easing.sharp,
          duration: theme.transitions.duration.leavingScreen,
        }),
        width: theme.spacing(7),
        [theme.breakpoints.up('sm')]: {
          width: theme.spacing(9),
        },
```

```
    }),
  },
 }),
);
        const defaultTheme = createTheme();
export default function Dashboard() {
 const [open, setOpen] = React.useState(true);
 const toggleDrawer = () => {
  setOpen(!open);
 };
          return (
  <ThemeProvider theme={defaultTheme}>
   <Box sx={{ display: 'flex' }}>
    <CssBaseline />
    <AppBar position="absolute" open={open}>
     <Toolbar
      sx={{
       pr: '24px', // keep right padding when drawer closed
      }}
     >
      <IconButton
       edge="start"
       color="inherit"
       aria-label="open drawer"
       onClick={toggleDrawer}
       sx={{
        marginRight: '36px',
```

```
        ...(open && { display: 'none' }),
      }}
    >
      <MenuIcon />
    </IconButton>
    <Typography
      component="h1"
      variant="h6"
      color="inherit"
      noWrap
      sx={{ flexGrow: 1 }}
    >
      Dashboard
    </Typography>
    <IconButton color="inherit">
      <Badge badgeContent={4} color="secondary">
        <NotificationsIcon />
      </Badge>
    </IconButton>
  </Toolbar>
</AppBar>
<Drawer variant="permanent" open={open}>
  <Toolbar
    sx={{
      display: 'flex',
      alignItems: 'center',
      justifyContent: 'flex-end',
```

```
      px: [1],
    }}
  >
    <IconButton onClick={toggleDrawer}>
      <ChevronLeftIcon />
    </IconButton>
  </Toolbar>
  <Divider />
  <List component="nav">
    {mainListItems}
    <Divider sx={{ my: 1 }} />
    {secondaryListItems}
  </List>
</Drawer>
<Box
  component="main"
  sx={{
    backgroundColor: (theme) =>
      theme.palette.mode === 'light'
        ? theme.palette.grey[100]
        : theme.palette.grey[900],
    flexGrow: 1,
    height: '100vh',
    overflow: 'auto',
  }}
>
  <Toolbar />
```

```
<Container maxWidth="lg" sx={{ mt: 4, mb: 4 }}>
  <Grid container spacing={3}>
    {/* Chart */}
    <Grid item xs={12} md={8} lg={9}>
      <Paper
        sx={{
          p: 2,
          display: 'flex',
          flexDirection: 'column',
          height: 240,
        }}
      >
        <Chart />
      </Paper>
    </Grid>
    {/* Recent Deposits */}
    <Grid item xs={12} md={4} lg={3}>
      <Paper
        sx={{
          p: 2,
          display: 'flex',
          flexDirection: 'column',
          height: 240,
        }}
      >
        <Deposits />
      </Paper>
```

```
            </Grid>
            {/* Recent Orders */}
            <Grid item xs={12}>
              <Paper sx={{ p: 2, display: 'flex', flexDirection: 'column' }}>
                <Orders />
              </Paper>
            </Grid>
          </Grid>
          <Copyright sx={{ pt: 4 }} />
        </Container>
      </Box>
    </Box>
  </ThemeProvider>
 );
}
```

CHAPTER 7

SCREEN SHOTS

1. Dashboard Page

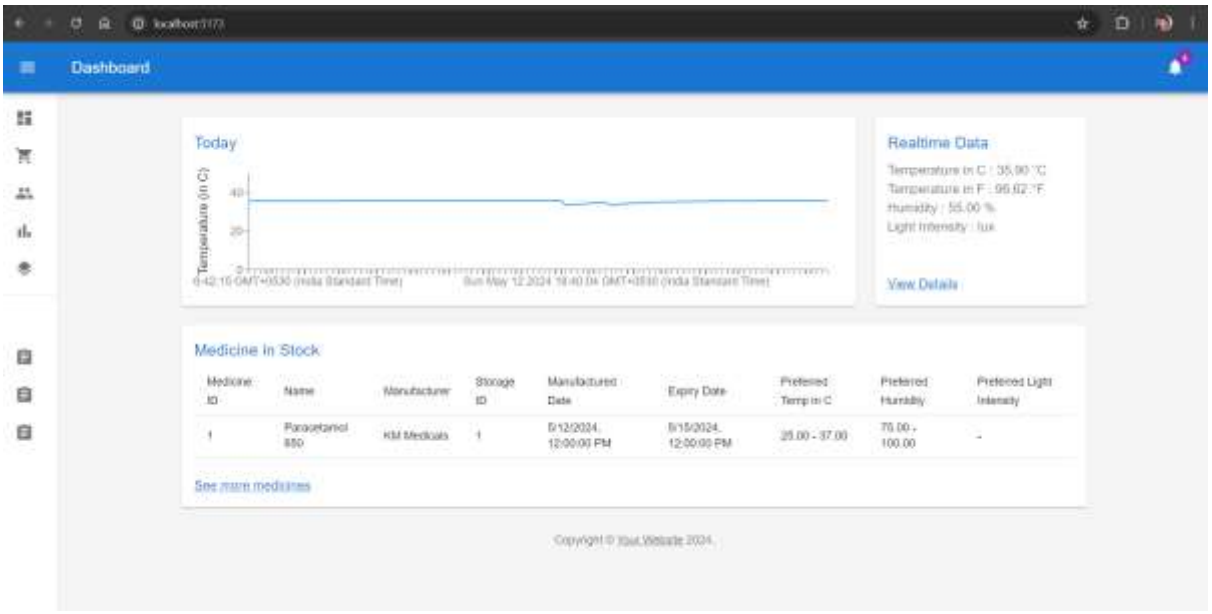

Figure 7.1 Responsive Dashboard

2. Data Sent from ESP8266-12E Nodemcu to Server

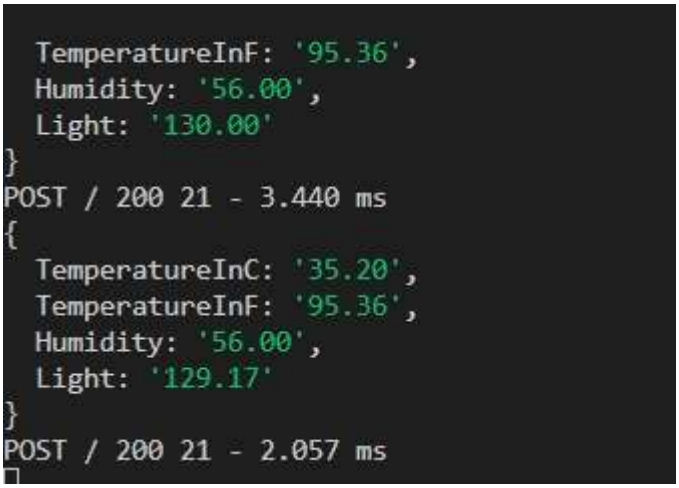

Figure 7.2 Data Received by Server from ESP8266-12E Nodemcu

CHAPTER 8

CONCLUSION AND FUTURE ENHANCEMENT

The pharmaceutical storage monitoring system with its real-time dashboard web application stands as a revolutionizing advancement in pharmaceutical storage management, ensuring the integrity and efficacy of stored medications. Through the integration of hardware sensors, data processing algorithms, and a user-friendly interface, the system provides stakeholders with unprecedented insight into storage conditions, enabling swift detection and response to deviations. Further improvement in pharmaceutical storage facilities signifies a paradigm shift in storage management practices, promising enhanced operational efficiency and risk mitigation.

Future enhancements can be directed towards several key areas. Enhanced data analytics techniques can unlock deeper insights into storage condition trends, informing proactive management strategies. Predictive maintenance algorithms can be developed to anticipate equipment failures and ensure uninterrupted monitoring operations. Integration with inventory management systems can streamline product tracking and monitor supply chains. Expanding the web application's capabilities for remote monitoring and control can provide stakeholders with greater flexibility and accessibility. Additionally, the integration of additional IoT devices and sensors, coupled with Artificial Intelligence, can enhance the system's intelligence and adaptability, ensuring compliance with evolving regulatory standards and facilitating seamless adherence to compliance requirements.

REFERENCES

1. Shafaat, K., Hussain, A., Kumar, B., Hasan, R., Prabhat, P., & Yadav, V. (2013). An overview: storage of pharmaceutical products. World J Pharm Pharm Sci, 2(5), 2499-2515.

2. Khuluza, F., Chiumia, F. K., Nyirongo, H. M., Kateka, C., Hosea, R. A., & Mkwate, W. (2023). Temperature variations in pharmaceutical storage facilities and knowledge, attitudes, and practices of personnel on proper storage conditions for medicines in southern Malawi. Frontiers in Public Health, 11, 1209903.

3. Gbenga, B. L., & Taiwo, Y. (2015). Studies of the effect of storage conditions on some pharmaceutical parameters of powders and tablets. Dhaka University Journal of Pharmaceutical Sciences, 14(2), 147-151.

4. Nhan, P. P., & Hoa, N. K. (2013). Effect of light and storage time on vitamin E in pharmaceutical products. British journal of pharmacology and toxicology, 4(5), 176-180.