Journal of Intelligent Decision Support System (IDSS) 6 (3) (2023), pp. 178-190

Published by: Institute of Computer Science (IOCS)

Journal of Intelligent Decision Support System (IDSS)

Journal homepage: www.idss.iocspublisher.org

Prototype temperature monitoring system for medicine refrigeration in the pharmaceutical installation

Willy[1], Haryono[2], Handri Santoso[3], Ito Wasito[4]

[1-4] *Universitas Pradita*

Article Info

Article history:

Received Aug 27, 2023
Revised Sep 15, 2023
Accepted Sep 30, 2023

Keywords:

Healthcare,
Internet of Things,
Technologies.

ABSTRACT

Patient safety in a hospital is a healthcare service that is safe and non-harmful to patients. All components of healthcare services (doctors, nurses, and other health teams) in hospitals must be aware of and care about patient safety while in the hospital. One of the most important components is monitoring the temperature of the medicine refrigeration. Human resources or work overload often leads to the need for more monitoring of the temperature of medicine refrigeration. Therefore, IoT technology can be the solution to assist in monitoring the temperature of the medicine refrigerator. The method used in this study is observation. Based on the conducted research, it is proven that the IOT system for temperature monitoring can reduce missed temperature records. The results of this study indicate that the DHT22 sensors have good accuracy as they remain within the accuracy range of the room thermometer used as a reference, with a temperature reading accuracy of ±1ºC and a maximum temperature measurement limit of 70ºC. The data collection process uses the ESP8266 as the microcontroller, which is then connected to the DHT22 module as a temperature and humidity sensor and sends adatabase every 30 seconds. The real-time temperature and humidity measurement results can be viewed through mobile apps using the Flutter programming language and the website. If the temperature exceeds 8ºC, the fan LED will automatically turn on and send notifications to WhatsApp registered using Python and Twilio. Furthermore, the existing data can be analyzed using a machine learning model, enabling the prediction of when the refrigerator will be damaged as a preventive measure.

Corresponding Author:

Willy,
Praditya University
Master of Information Technology, Faculty of Science and Technology, Pradita University
Scientia Business Park, Jl. Gading Serpong Boulevard, Tangerang Regency, Banten, Indonesia 15810
Email: willy@student.pradita.ac.id

Introduction

The safety of patients has become an international and national issue for hospitals, which is an important component of quality service health, the basic principle of service to patients, and the critical component of quality management (Adhi & Ningsih, 2020; Kristianto, 2019). Patients, as a user of healthcare services, obtain security and safety while in treatment at the hospital (Adhi & Ningsih, 2020; Simanjuntak et al., 2020). The International Patient Safety goal is one of the standards within the Joint Commission International (JCI) accreditation that aims to make nursing care provided in hospitals safer (Jannah, 2020; JCI, 2017; Kalsoom et al., 2022; Nurhanifah et al., 2021; The Joint Commission International, 2017).

According to Nuryanti (2019), patient safety in hospitals is a healthcare service that is safe and non-harmful to patients. All healthcare service components (doctors, nurses, and other healthcare teams) in hospitals must be aware of and concerned about patient safety while they are in the hospital (Irwanti et al., 2022; Kalsoom et al., 2022). JCI has determined the goals of patient safety in hospitals (*International et al./IPSG*) to improve patient safety, namely 1) Correctly identifying patients; 2) Improving effective communication; 3) Using safe medication; 4) Ensuring the right site, the right procedures, and appropriate patient; 5) Reducing the risk infection related healthcare service and identifying the risk of patients fall (Joshi & Saini, 2022; Kurniawan, 2020).

Implementation of safety patients aims to reduce the percentage of unexpected events *(KTD),* which often occur in patients during hospitalization (Kamalia, 2022; Kurniawan, 2020; Saryadi, 2022). Consequently, this situation greatly disadvantages the patients themselves and the hospital as a whole. Individual negligence poses a threat to unexpected incidents occurring in patients. Approximately 100,000 people die each year due to medical errors from medication side effects, surgical complications, system errors, and treatment mistakes (Herikurniawan, 2020; Kalsoom et al., 2022). Based on these reports, improving the patient safety programs is crucial, particularly regarding NPSG 3, which pertains to safe medication use. One effort to maintain the quality of drugs in hospitals is by ensuring proper medication storage temperature. By the regulations of the Ministry of Health (Permenkes) number 7 of 2019, temperature and air humidity in specialized areas must be regularly monitored every day and substantiated with monitoring reports. The Indonesian Pharmacopoeia Edition VI 2020 has established temperature guidelines to achieve medical stability that meets the criteria. This directive applies to all medications, except when the label on the medication indicates different storage temperatures based on stability studies conducted by the pharmaceutical industry (Herikurniawan, 2020; Kobayashi et al., 2021).

Examples of medicines that must be stored in the refrigerator are drug insulin and injection types (Kumar et al., 2022; Li et al., 2018; Sukmawati E. et al., 2018). Technology is important to implement and achieve patient safety (Hughes-Lartey et al., 2021; Humayun et al., 2021; Wiwoho Mudjanarko et al., 2017). Not all hospitals have sufficient storage space for advanced medicines, such as Metro Hospitals M. Toha. There is a need for a tool that can connect to the Internet to facilitate temperature monitoring and recording in the pharmacy unit more easily, effectively, and in real time. This concept is known as the Internet of Things (IoT). This research aims to discover the role of IOT, particularly in the pharmacy unit, for real-time, paperless, and automated temperature monitoring of refrigeration units such as Metro Hospital M. Toha.

Method

ESP8266 was connected to the PC as a source/*power supply,* then to the DHT22 sensor and set in Arduino IDE. If the temperature exceeds 8^{o}C, the LED and L9110 fan will be turned on until the temperature meets standard criteria. Meanwhile, ESP8266 will send data to the web server and can be displayed on mobile apps in 30 seconds. Temperature and humidity can be seen in a mareal-timeh mobile apps .

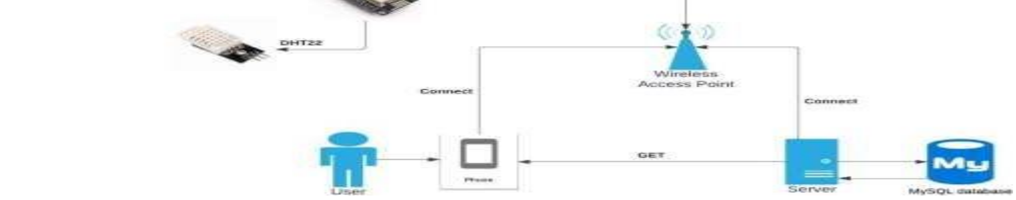


Figure 1 Method of Collecting Data

Results and Discussions

Comparison Between Results Measurement and Refrigerator

This chapter will elaborate on the research findings and discuss the study results. The research findings are presented in the form of tabla e, which summarizes the research findings. The table is displayed to help readers comprehend research findings. These are obtained by comparing the DHT22 with a refrigerator to determine the differences between the two devices.

Table 1. Comparison Between DHT22 and Refrigerator

Time Measurement	Results Measurement Temperature		Difference
	DHT22(◦C)	Refrigerator(%)	
19 Apr 2023 10.52.45	6,1	5,9	0,2
19 Apr 2023 10.53.23	6,1	5,9	0,2
19 Apr 2023 10.54.09	6,1	5,9	0,2
19 Apr 2023 10.54.41	6,1	5,9	0,2
19 Apr 2023 10.55.56	6,1	5,9	0,2
19 Apr 2023 10.56.34	6,1	5,9	0,2
19 Apr 2023 10.57.06	6,1	5,9	0,2
19 Apr 2023 10.58.12	6,2	5,9	0,3
19 Apr 2023 10.58.47	6,1	6,0	0,1
19 Apr 2023 10.59.20	6,1	6,0	0,1
19 Apr 2023 10.59.55	6,1	6,0	0,1
19 Apr 2023 11.01.09	6,2	6,0	0,2
19 Apr 2023 11.01.46	6,1	6,0	0,1
19 Apr 2023 11.02.22	6,1	6,0	0,1
19 Apr 2023 11.02.56	6,2	6,0	0,2
19 Apr 2023 11.03.30	6,1	6,0	0,1
19 Apr 2023 11.04.10	6,1	6,0	0,1
19 Apr 2023 11.05.23	6,2	6,0	0,2
19 Apr 2023 11.05.59	6,1	6,0	0,1
19 Apr 2023 11.06.39	6,1	6,0	0,1
19 Apr 2023 11.07.14	6,1	6,0	0,1
19 Apr 2023 11.07.52	6,1	6,0	0,1
19 Apr 2023 11.08.27	6,1	6,0	0,1
19 Apr 2023 11.09.00	6,2	6,0	0,2
19 Apr 2023 11.09.33	6,1	6,0	0,1
19 Apr 2023 11.10.06	6,1	6,0	0,1
19 Apr 2023 11.10.38	6,1	6,0	0,1
19 Apr 2023 11.11.14	6,1	5,9	0,2
19 Apr 2023 11.11.54	6,1	5,9	0,2
19 Apr 2023 11.12.33	6,1	5,9	0,2
19 Apr 2023 11.13.07	6,1	5,9	0,2
19 Apr 2023 11.14.18	6,1	5,9	0,2
19 Apr 2023 11.14.54	6,1	5,9	0,2
19 Apr 2023 11.15.30	6,1	5,9	0,2
19 Apr 2023 11.16.09	6,1	5,9	0,2
19 Apr 2023 11.16.45	6,1	5,9	0,2
19 Apr 2023 11.17.40	6,1	5,9	0,2
19 Apr 2023 11.18.08	6,1	5,9	0,2
19 Apr 2023 11.18.42	6,1	5,9	0,2
19 Apr 2023 11.19.54	6,2	5,9	0,3
Flat- flat deviation whole			0,1675

Formulas :
e = [$-b$]

Description:
e = Error
a = Sensors
DHT22b = Refrigerator

Based on the table above, the collected sample data shows a difference between the refrigerator and DHT22 of 0,3 or an average of 0,1675. This shows that the DHT22 sensor has good accuracy because, according to the sensor data sheet, the measured temperature should be within the range of -40°C with a maximum temperature measurement limit of 80°C and accuracymeasurement ± 0.5°C.

Measurement Results

Table 2. Measurement with DHT22

Time measurement	**Measurement**		
	Temperature (°C)	**Humidity(%)**	**Status**
19 Apr 2023 08.56.37	6.9	81	Sent
19 Apr 2023 08.57.09	6.9	81	Sent
19 Apr 2023 08.57.41	6.9	81	Sent
19 Apr 2023 08.58.14	6.9	81	Sent
19 Apr 2023 08.58.46	6.8	82	Sent
19 Apr 2023 08.59.18	6.8	82	Sent
19 Apr 2023 08.59.50	6.8	82	Sent
19 Apr 2023 09.00.22	6.8	82	Sent
19 Apr 2023 09.00.55	6.7	82	Sent
19 Apr 2023 09.01.36	6.7	82	Sent
19 Apr 2023 09.02.07	6.7	83	Sent
19 Apr 2023 09.02.39	6.7	83	Sent
19 Apr 2023 09.03.11	6.7	83	Sent
19 Apr 2023 09.03.43	6.7	83	Sent
19 Apr 2023 09.04.16	6.7	83	Sent
19 Apr 2023 09.05.27	6.6	83	Sent
19 Apr 2023 09.06.00	6.7	85	Sent
19 Apr 2023 09.06.32	6.8	85	Sent
19 Apr 2023 09.08.21	6.7	87	Sent
19 Apr 2023 09.08.38	6.9	87	Sent
19 Apr 2023 09.08.59	6.8	87	Sent
19 Apr 2023 09.09.32	6.8	87	Sent
19 Apr 2023 09.10.25	6.8	88	Sent
19 Apr 2023 09.11.23	6.7	88	Sent
19 Apr 2023 09.11.34	6.7	88	Sent
19 Apr 2023 09.12.08	6.7	88	Sent
19 Apr 2023 09.12.44	6.7	88	Sent
19 Apr 2023 09.13.57	6.6	88	Sent
19 Apr 2023 09.16.03	6.6	89	Sent
19 Apr 2023 09.17.40	6.6	89	Sent
19 Apr 2023 09.18.15	6.6	89	Sent
19 Apr 2023 09.18.50	6.6	89	Sent
19 Apr 2023 09.19.25	6.6	89	Sent
19 Apr 2023 09.20.35	6.6	89	Sent
19 Apr 2023 09.21.54	6.6	89	Sent
19 Apr 2023 09.21.55	6.6	89	Sent
19 Apr 2023 09.22.22	6.6	89	Sent
19 Apr 2023 09.23.00	6.6	89	Sent
19 Apr 2023 09.23.33	6.6	89	Sent
19 Apr 2023 09.24.11	6.6	90	Sent

Based on the table above, from the collected temperature measurement data, the captured data taken by DHT22 is 100%, set to data the base during the active internet connection, and there is no change on SinD and WiFi passwords.

Damage Predictions of Refrigerator with Flutter

Based on data taken for eight days in three different refrigerators, the description can be seen as follows:

Table 3. Total of Datasets

No	Storage/Brand		***n* data**
1	GEA Expo		10.315 data
2	RSA 1		1.992 data
3	RSA 2		13.542 data
		Total	25.849 data

Description:
n data = Total obtained data (data can be seen on attachment of data measurement)

Damage Predictions of Refrigerators with Machine Learning

The prediction was made by taking data for eight days and analyzing each refrigerator or storage.

Table 3. Datasets

Variable	Description
Date	Date
Temp	Temperature
Storage	*Storage*/Refrigerator

a. Refrigerator 1 – Gea Expo (Medical Warehouse)

1. Load data sensors on refrigerator/storage 1, along with libraries needed.

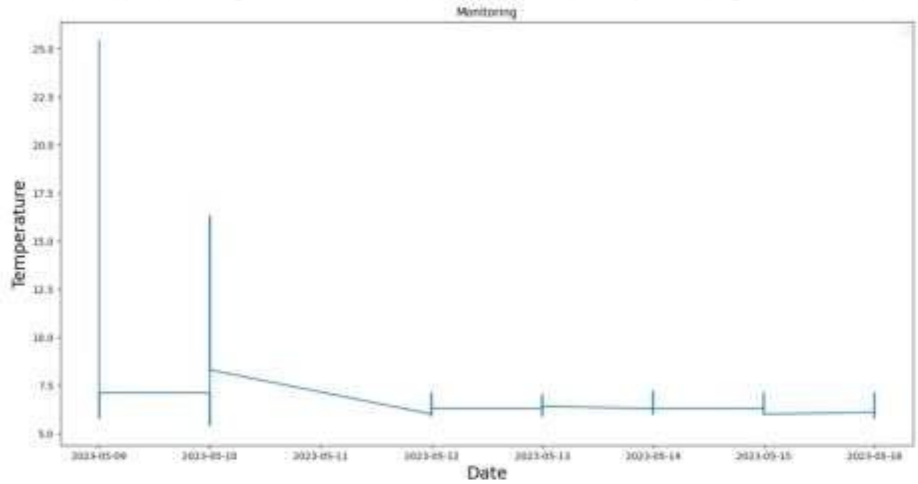
```
import pandas as pd
import xgboost as xgb
import matplotlib.pyplot as plt


df = pd.read_excel('sensor_1.xlsx', usecols=[0,2,3,4])
df.head()
```

	Date	Temp	Hum	Store
0	2023-05-09	25.4	60	1
1	2023-05-09	25.2	65	1
2	2023-05-09	20.9	49	1
3	2023-05-09	19.3	53	1
4	2023-05-09	17.9	55	1

```
plt.figure(figsize=(16,8))
plt.title("Monitoring")
plt.plot(df['Date'], df['Temp'])
plt.xlabel('Date', fontsize=18)
plt.ylabel('Temperature', fontsize=18)
plt.legend()
plt.show()
```

2. Display data in visual form to make it easier to conduct analysis.

```
WARNING:matplotlib.legend:No artists with labels found to put in legend.  Note that
```

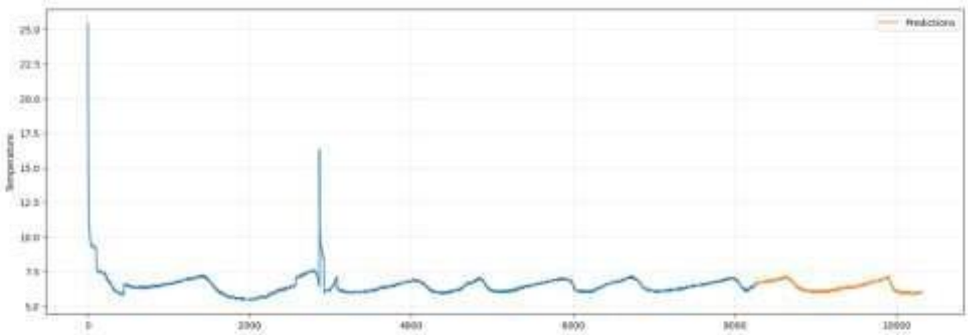


3. Create training data and testing data (80%), then convert the type of data objects to int, then load libraries XGBoost.

```
train_data = df.iloc[:int(.80 * len(df)) , :]
test_data = df.iloc[int(.80* len(df)):, :]

df['Date'] = df['Date'].astype('int')

df.info()

    <class 'pandas.core.frame.DataFrame'>
    RangeIndex: 10315 entries, 0 to 10314
    Data columns (total 4 columns):
     #   Column  Non-Null Count  Dtype
    ---  ------  --------------  -----
     0   Date    10315 non-null  int64
     1   Temp    10315 non-null  float64
     2   Hum     10315 non-null  int64
     3   Store   10315 non-null  int64
    dtypes: float64(1), int64(3)
    memory usage: 322.5 KB

features = ['Hum','Temp']
target = 'Temp'

model = xgb.XGBRegressor()
model.fit(train_data[features], train_data[target])

    XGBRegressor
    XGBRegressor(base_score=None, booster=None, callbacks=None,
                 colsample_bylevel=None, colsample_bynode=None,
                 colsample_bytree=None, early_stopping_rounds=None,
                 enable_categorical=False, eval_metric=None, feature_types=None,
                 gamma=None, gpu_id=None, grow_policy=None, importance_type=None,
                 interaction_constraints=None, learning_rate=None, max_bin=None,
                 max_cat_threshold=None, max_cat_to_onehot=None,
                 max_delta_step=None, max_depth=None, max_leaves=None,
                 min_child_weight=None, missing=nan, monotone_constraints=None,
                 n_estimators=100, n_jobs=None, num_parallel_tree=None,
                 predictor=None, random_state=None, ...)
```

4. Conduct prediction and visualize in chart line.

```
predictions = model.predict(test_data[features])
print("Model Predictions : ")
print(predictions)

    Model Predictions :
    [6.500081 6.500081 6.500081 ... 6.000044 6.000044 6.000044]

print("Actual Values : ")
print(test_data[target])

    Actual Values :
    8252     6.5
    8253     6.5
    8254     6.5
    8255     6.6
    8256     6.6
            ...
    10310    6.0
    10311    6.0
    10312    6.0
    10313    6.0
    10314    6.0
    Name: Temp, Length: 2063, dtype: float64

plt.figure(figsize=(18, 6))
plt.grid(color='darkgray', linestyle=':', linewidth=0.5)
plt.plot(df['Temp'])
# plt.plot(test_data[target].index, predictions, label = 'Predictions')
plt.plot(test_data[target].index, predictions, label = 'Predictions')
plt.ylabel('Temperature')
plt.legend()
plt.show()
```



Based on the data above, the prediction results for refrigerator storage 1 showed quite good results, in the range of 5°C - 7.5°C. The potential date of the storage failure can only be determined once the required data is sufficient.

b. Refrigerator 2 – RSA (Outpatient Pharmacy Installation)

1. Load sensor data on refrigerator/storage two and the required libraries.

```
import pandas as pd
import xgboost as xgb
import matplotlib.pyplot as plt
```

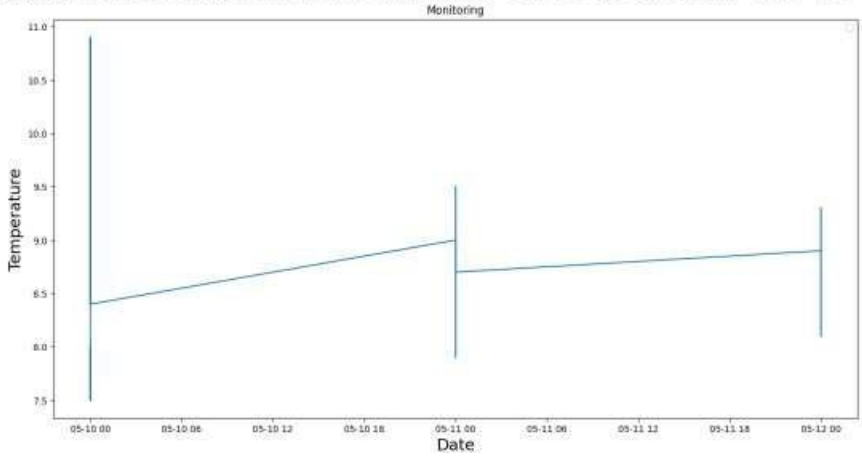
```
df = pd.read_excel('sensor_2.xlsx', usecols=[0,2,3,4])
df.head()
```

	Date	Temp	Hum	Store
0	2023-05-10	8.0	82	2
1	2023-05-10	7.9	82	2
2	2023-05-10	7.9	82	2
3	2023-05-10	8.0	82	2
4	2023-05-10	8.0	82	2

```
plt.figure(figsize=(16,8))
plt.title("Monitoring")
plt.plot(df['Date'], df['Temp'])
plt.xlabel('Date', fontsize=18)
plt.ylabel('Temperature', fontsize=18)
plt.legend()
plt.show()
```

2. Display data in the visual form to make it easier to conduct analysis.

```
WARNING:matplotlib.legend:No artists with labels found to put in legend.  Note that
```



3. Create training data and testing data (80%), then convert the type of data objects to int, then load libraries XGBoost.

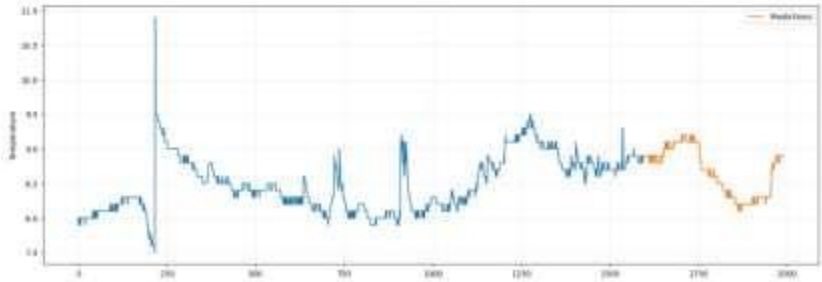
```
train_data = df.iloc[:int(.80 * len(df)) , :]
test_data = df.iloc[int(.80* len(df)):, :]

df['Date'] = df['Date'].astype('int')

df.info()

    <class 'pandas.core.frame.DataFrame'>
    RangeIndex: 1992 entries, 0 to 1991
    Data columns (total 4 columns):
     #   Column  Non-Null Count  Dtype
    ---  ------  --------------  -----
     0   Date    1992 non-null   int64
     1   Temp    1992 non-null   float64
     2   Hum     1992 non-null   int64
     3   Store   1992 non-null   int64
    dtypes: float64(1), int64(3)
    memory usage: 62.4 KB
```

4. Conduct prediction and visualize in chart line.

```
predictions = model.predict(test_data[features])
print("Model Predictions : ")
print(predictions)
```

```
Model Predictions :
[8.800186 8.899822 8.899822 8.899822 8.899822 8.899822 8.899822 8.899822
 8.899822 8.899822 8.899822 8.899822 8.899822 8.899822 8.899822 8.899822
 8.899822 8.899822 8.899822 8.800186 8.800186 8.899822 8.899822 8.899822
 8.800186 8.800186 8.899822 8.899822 8.899822 8.899822 8.800186 8.899822
 8.899822 8.899822 8.899822 8.899822 8.899822 8.899822 8.800186 8.800186
 8.899822 8.800186 8.800186 8.800186 8.800186 8.800186 8.800186 8.800186
 8.800186 8.800186 8.800186 8.899822 8.800186 8.899822 8.800186 8.899822
 8.899822 8.899822 8.899822 8.899822 9.000281 9.000281 9.000281 9.100008
 9.100008 9.000281 9.100008 9.000281 9.100008 9.100008 9.100008 9.000281
 9.100008 9.000281 9.100008 9.000281 9.000281 9.000281 9.000281 9.000281
 9.000281 9.000281 9.000281 9.000281 9.000281 9.000281 9.100008 9.000281
 9.100008 9.100008 9.100008 9.100008 9.100008 9.100008 9.100008 9.100008
 9.100008 9.100008 9.100008 9.100008 9.100008 9.100008 9.100008 9.100008
 9.100008 9.100008 9.100008 9.100008 9.100008 9.100008 9.200009 9.100008
 9.100008 9.100008 9.200009 9.200009 9.200009 9.200009 9.200009 9.200009
 9.200009 9.200009 9.200009 9.200009 9.100008 9.100008 9.100008 9.100008
 9.100008 9.100008 9.100008 9.100008 9.100008 9.100008 9.200009 9.200009
 9.100008 9.100008 9.100008 9.100008 9.200009 9.100008 9.100008 9.100008
 9.200009 9.100008 9.100008 9.100008 9.100008 9.100008 9.100008 9.100008
 9.100008 9.100008 9.100008 9.100008 9.100008 9.100008 9.100008 9.000281
 9.000281 9.000281 8.899822 8.899822 8.800186 8.699894 8.699894 8.699894
 8.699894 8.699894 8.699894 8.699894 8.699894 8.699894 8.699894 8.699894
 8.699894 8.699894 8.699894 8.699894 8.699894 8.699894 8.699894 8.699894
 8.699894 8.699894 8.699894 8.600369 8.699894 8.699894 8.699894 8.600369
 8.699894 8.600369 8.600369 8.600369 8.600369 8.600369 8.600369 8.600369
 8.600369 8.600369 8.699894 8.600369 8.600369 8.600369 8.600369 8.600369
 8.600369 8.600316 8.600316 8.499831 8.499831 8.499831 8.499831 8.499831
 8.499831 8.499778 8.499831 8.499778 8.499778 8.499778 8.499778 8.499778
 8.499778 8.499778 8.400107 8.400107 8.400107 8.400107 8.400107 8.400107
 8.400107 8.400107 8.400107 8.299921 8.299921 8.299921 8.299921 8.299921
 8.299921 8.400107 8.299921 8.299921 8.299921 8.400107 8.299921 8.299921
 8.400107 8.299921 8.299921 8.299921 8.299921 8.299921 8.299921 8.299921
 8.299921 8.299921 8.299921 8.299921 8.299921 8.200088 8.200088 8.200088
 8.200088 8.200088 8.200088 8.200088 8.200088 8.200088 8.099905 8.200088
 8.099905 8.200088 8.099933 8.200088 8.099933 8.099905 8.099933 8.200046
8.200088 8.200088 8.200088 8.200088 8.200088 8.200088 8.200088 8.200088
8.200088 8.200088 8.200088 8.200088 8.200088 8.200088 8.200088 8.200088
8.200088 8.200088 8.200088 8.200088 8.200088 8.200088 8.200088 8.200088
8.200088 8.200088 8.200088 8.200088 8.200088 8.200088 8.299921 8.299921
8.299921 8.200088 8.299921 8.200088 8.200088 8.200088 8.200088 8.200088
8.200088 8.299921 8.299921 8.200088 8.299921 8.299921 8.299921 8.299921
8.299921 8.299921 8.299921 8.299921 8.299921 8.299921 8.299921 8.299921
8.299921 8.299921 8.299921 8.299921 8.299921 8.299921 8.299921 8.299921
8.299921 8.299921 8.299921 8.200088 8.299921 8.299921 8.299921 8.299921
8.299921 8.299921 8.299921 8.299921 8.299921 8.299921 8.299921 8.299921
8.299921 8.400107 8.499778 8.699841 8.699841 8.699841 8.699841 8.800133
8.800133 8.699841 8.800133 8.800133 8.800133 8.800133 8.800133 8.800133
8.800133 8.800133 8.899769 8.899769 8.899769 8.899769 8.899769 8.800133
8.899769 8.800133 8.899769 8.899769 8.899769 8.899769 8.899769 8.899769
8.899769 8.899769 8.899769 8.899769 8.899769 8.899769 8.899769]
```

```
plt.figure(figsize=(18, 6))
plt.grid(color='darkgray', linestyle=':', linewidth=0.5)
plt.plot(df['Temp'])
# plt.plot(test_data[target].index, predictions, label = 'Predictions')
plt.plot(test_data[target].index, predictions, label = 'Predictions')
plt.ylabel('Temperature')
plt.legend()
plt.show()
```



Based on the data above, it can be seen that the prediction results for refrigerators/storage two show that the storage is in bad condition. This is evident from the prediction results, which show temperatures in the range of 8.5ºC - 9ºC in the near future, followed by a return to the normal temperature of 8ºC. Then, there will be another increase, reaching up to 9ºC.

C. Refrigerator 3 – RSA (Inpatient Pharmacy Installation)

1. *Load* data sensors on refrigerator/*storage,* along with the required libraries.

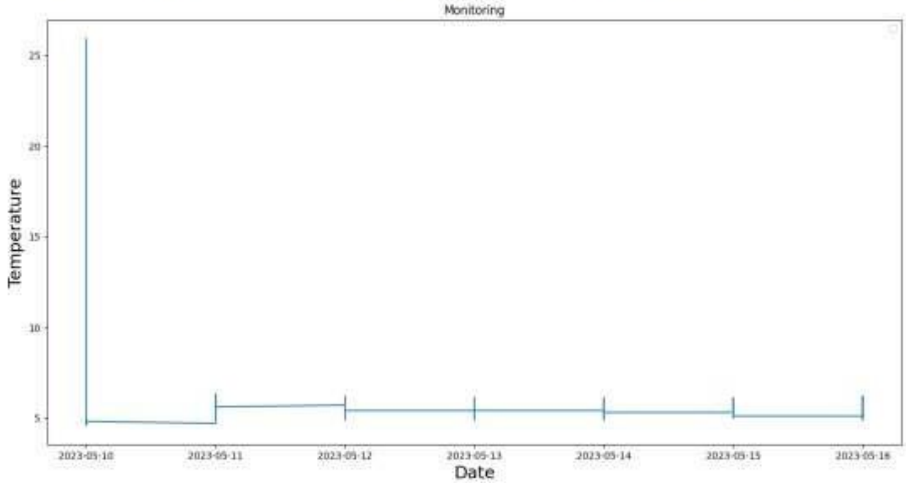
```
import pandas as pd
import xgboost as xgb
import matplotlib.pyplot as plt

df = pd.read_excel('sensor_3.xlsx', usecols=[0,2,3,4])
df.head()
```

	Date	Temp	Hum	Store
0	2023-05-10	25.9	68	3
1	2023-05-10	18.7	63	3
2	2023-05-10	15.9	66	3
3	2023-05-10	14.4	68	3
4	2023-05-10	13.5	68	3

```
plt.figure(figsize=(16,8))
plt.title("Monitoring")
plt.plot(df['Date'], df['Temp'])
plt.xlabel('Date', fontsize=18)
plt.ylabel('Temperature', fontsize=18)
plt.legend()
plt.show()
```

2. Display data in the visual form to make it easier to conduct analysis.

```
WARNING:matplotlib.legend:No artists with labels found to put in legend.  Note that
```

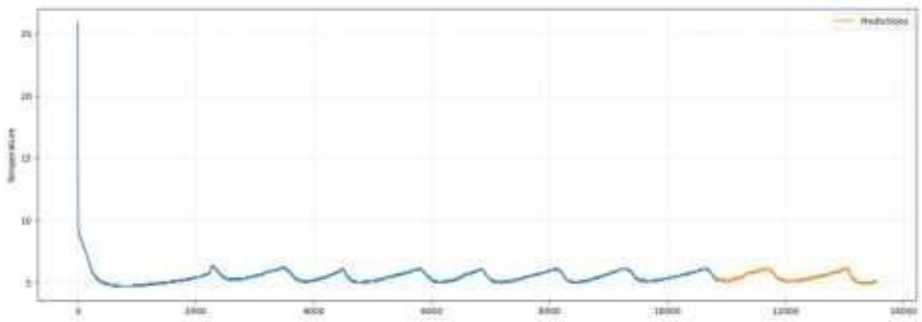


3. Create training data and testing data (80%), then convert the type of data objects to int, then load libraries XGBoost.

```
train_data = df.iloc[:int(.80 * len(df)) , :]
test_data = df.iloc[int(.80* len(df)):, :]

df['Date'] = df['Date'].astype('int')

df.info()

    <class 'pandas.core.frame.DataFrame'>
    RangeIndex: 13542 entries, 0 to 13541
    Data columns (total 4 columns):
     #   Column  Non-Null Count  Dtype
    ---  ------  --------------  -----
     0   Date    13542 non-null  int64
     1   Temp    13542 non-null  float64
     2   Hum     13542 non-null  int64
     3   Store   13542 non-null  int64
    dtypes: float64(1), int64(3)
    memory usage: 423.3 KB
```

4. Conduct prediction and visualize in chart line.

```
predictions = model.predict(test_data[features])
print("Model Predictions : ")
print(predictions)

    Model Predictions :
    [5.199977 5.199977 5.199977 ... 5.100006 5.100006 5.100006]

print("Actual Values : ")
print(test_data[target])

    Actual Values :
    10833    5.2
    10834    5.2
    10835    5.2
    10836    5.2
    10837    5.2
            ...
    13537    5.0
    13538    5.0
    13539    5.1
    13540    5.1
    13541    5.1
    Name: Temp, Length: 2709, dtype: float64

plt.figure(figsize=(18, 6))
plt.grid(color='darkgray', linestyle=':', linewidth=0.5)
plt.plot(df['Temp'])
# plt.plot(test_data[target].index, predictions, label = 'Predictions')
plt.plot(test_data[target].index, predictions, label = 'Predictions')
plt.ylabel('Temperature')
plt.legend()
plt.show()
```



Based on the data above, the prediction results for refrigerators/storage 3 showed quite good results, still within the range of 5^{o}C - 7.5^{o}C. The potential date of storage malfunction can only be determined once the necessary data is sufficient.

Conclusions

IOT technology can help the staff monitor the temperature of the medicine refrigerator in the pharmacy installation. The current study is limited in measuring temperature and humidity. Future research, it is expected to measure other parameters as well.

References

Adhi, S. N., & Ningsih, K. P. (2020). Manajemen Data Standar Pelayanan Minimal Rumah Sakit. *Jurnal Rekam Medis Dan Informasi Kesehatan*, *3*(2). https://doi.org/10.31983/jrmik.v3i2.6288

Herikurniawan. (2020). Patient safety and international patient safety goals (IPSGS) in the field of respirology. *Indonesian Journal of Chest*, *7*(1).

Hughes-Lartey, K., Li, M., Botchey, F. E., & Qin, Z. (2021). Human factor, a critical weak point in the information security of an organization's Internet of things. *Heliyon*, *7*(3). https://doi.org/10.1016/j.heliyon.2021.e06522

Humayun, M., Jhanjhi, N. Z., Alsayat, A., & Ponnusamy, V. (2021). Internet of things and ransomware: Evolution, mitigation and prevention. In *Egyptian Informatics Journal* (Vol. 22, Issue 1). https://doi.org/10.1016/j.eij.2020.05.003

Irwanti, F., Guspianto, G., Wardiah, R., & Solida, A. (2022). Hubungan Komunikasi Efektif dengan Pelaksanaan Budaya Keselamatan Pasien di RSUD Raden Mattaher Provinsi Jambi. *Jurnal Kesmas Jambi*, *6*(1), 32–41. https://doi.org/10.22437/jkmj.v6i1.15551

Jannah, F. (2020). Knowledge of Nurses about Third Goal International Patients Safety Goal. *JPK : Jurnal Proteksi Kesehatan*, *9*(1). https://doi.org/10.36929/jpk.v9i1.239

JCI. (2017). International Patient Safety Goals (IPSGs). *The Joint Commission International*.

Joshi, D. C., & Saini, R. S. (2022). Awareness and Compliance to International Patient Safety Goals among Healthcare Personnel of a Tertiary Care Hospital in Northern India. *JOURNAL OF CLINICAL AND DIAGNOSTIC RESEARCH*. https://doi.org/10.7860/jcdr/2022/58145.17113

Kalsoom, Z., Victor, G., Virtanen, H., & Sultana, N. (2022). What really matters for patient safety: Correlation of nurse competence with international patient safety goals. *Journal of Patient Safety and Risk Management*. https://doi.org/10.1177/25160435221133955

Kamalia, L. O. (2022). Manajemen Pelayanan Rumah Sakit dan Puskesmas. *Book*, *490*(Januari).

Kobayashi, K., Ando, K., Nakashima, H., Machino, M., Kanbara, S., Ito, S., Inoue, T., Yamaguchi, H., Ishiguro, N., & Imagama, S. (2021). Challenges for Joint Commission International accreditation: performance of orthopedic surgeons based on International Patient Safety Goals. *Nagoya Journal of Medical Science*, *83*(1). https://doi.org/10.18999/nagjms.83.1.87

Kristianto, Y. (2019). Pengembangan Sistem Informasi Manajemen. *Seminar Nasional Pascasarjana 2019*.

Kumar, R. L., Khan, F., Kadry, S., & Rho, S. (2022). A Survey on blockchain for industrial Internet of Things: Blockchain for Internet of Things. *Alexandria Engineering Journal*, *61*(8). https://doi.org/10.1016/j.aej.2021.11.023

Kurniawan, H. (2020). Patient safety and international patient safety goals (IPSGS) in the field of respirology. *Indonesian Journal of Chest*, *7*(1).

Li, S., Xu, L. Da, & Zhao, S. (2018). 5G Internet of Things: A survey. In *Journal of Industrial Information Integration* (Vol. 10). https://doi.org/10.1016/j.jii.2018.01.005

Nurhanifah, N., Kamil, H., Syahrul, S., Marlina, M., & Marianthi, D. (2021). The Relationship between Nurse's Characteristics and Motivation with the Implementation of International Patient Safety Goal. *Media Karya Kesehatan*, *4*(2). https://doi.org/10.24198/mkk.v4i2.34665

Nuryanti. (2019). *Peran Self Efficacy Dalam Memediasi Pengaruh Persepsi Biaya Pendidikan, Kondisi Ekonomi Orang Tua, Dan Linhkungan Teman Sebaya Terhadap Minat Melanjutkan S2*. 1–235.

Saryadi, S. (2022). Organisasi dan Manajemen Pelayanan Kesehatan. In *Tata Mutiara Hidup Indonesia*.

Simanjuntak, M. R., Pasaribu, N. D., & Sibagariang, E. E. (2020). Manajemen sumber daya manusia dengan waktu tunggu pasien di pelayanan farmasi Rumah Sakit X. *Jurnal Prima Medika Sains*, *2*(2). https://doi.org/10.34012/jpms.v2i2.973

Sukmawati E, wahyunita yulia sari, & indah sulistyoningrum. (2018). *Farmakologi Kebidanan*. Trans Info Media (TIM). https://scholar.google.com/scholar?oi=bibs&cluster=11760770443894442881&btnI=1&hl=id

The Joint Commission International. (2017). International Patient Safety Goals (IPSGs). *The Joint Commission International*, *February 2018*.

Wiwoho Mudjanarko, S., Winardi, S., & Daniel Limantara, A. (2017). Pemanfaatan Internet of Things (Iot) Sebagai Solusi Manejemen Transportasi Kendaraan Sepeda Motor. *Seminar Nasional Aplikasi Teknologi Prasarana Wilayah X (ATPW)*, *August*.