Pre-symptomatic detection of Covid-19 system based on deep learning technology from smartwatch data

BY

Anon Suphatphon
Pattapon Tanankakorn

A PROJECT SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF BACHELOR OF ENGINEERING IN BIOMEDICAL ENGINEERING
KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG
ACADEMIC YEAR 2022

SCHOOL OF ENGINEERING

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

PROJECT CERTIFICATE

Project Title	Pre-symptomatic detection of Covid-19 system based on deep learning technology from smartwatch data
Student Name	Mr. Anon Suphatphon Student ID. 62011090 Mr. Pattapon Tanankakorn Student ID. 62011190
Degree	Bachelor of Engineering in Biomedical Engineering
Project Advisor	Signed: ____________________ (Assoc. Prof. Dr. Wibool Piyawattanametha)
Committee	Signed: ____________________ (Assoc. Prof. Dr. Chuchart Pintavirooj)
Committee	Signed: ____________________ (Asst. Prof. Dr. Treesukon Treebupachatsakul)
Committee	Signed: ____________________ (Asst. Prof. Dr. Kasama Srirussamee)
Head of Department	Signed: ____________________ (Asst. Prof. Dr. Sarinporn Visitsattapongse)

Project Title	Pre-symptomatic detection of Covid-19 system based on deep learning technology from smartwatch data
Student Name	Mr. Anon Suphatphon Mr. Pattapon Tanankakorn
Degree	Bachelor of Engineering in Biomedical Engineering
Project Advisor	Assoc. Prof. Dr. Wibool Piyawattanametha
Academic Years	2022

ABSTRACT

The rapid spread of the COVID-19 pandemic has highlighted the urgency of infectious disease surveillance and early detection. The purpose of this research is to develop a COVID-19 pre-symptomatic detection system using deep learning and data from smartwatches. The system makes use of watch technology to collect vital physiological information such as temperature, blood oxygen saturation, step count, and heart rate. These metrics are generated by a data collection technique that includes sensors and related apps found in smartwatches.

The data acquisition technique involves the employment of smartwatches with sensors to measure vital signs and critical values. The obtained data is then merged with artificial intelligence (AI) algorithms in a database to detect COVID-19 pre-symptomatic cases. Furthermore, a telehealth system is employed to facilitate communication between patients, smartwatches, and medical experts. The approach allows doctors to get vital signs as well as patient-reported concerns for examination and diagnosis.

Two smartwatch models, Fitbit and J-style, are utilized in this research. The J-style model requires the use of the Nordic Semiconductor's nRF Connect application to explore the Bluetooth Generic Attribute (GATT) services, characteristics exposed by the smartwatch and Kotlin programming language for develop application. On the other hand, the Fitbit model involves the development of a custom application using the Fitbit developer website.

The research also emphasizes the importance of data acquisition, telehealth systems, and collaboration between patients and healthcare professionals in managing crisis circumstances effectively. Two platform, mobile application and dashboard website, are utilized in this research. The website represents a comprehensive and visually appealing representation of the acquired information from smartwatches and facilitates about communication and collaboration between patients and doctors. Patients can securely share their data with their healthcare providers, enabling remote monitoring. Doctors can provide personalized recommendations, set goals, and track the progress of their patients.

In summary, the results of the machine learning model show poor performance for the anomaly detection task. Accuracy is 0.09, precision is 0.01, recall is 0.09, and F1 Scores is 0.03 score were all low, suggesting that the model could not correctly classify cases as anomalies. This suggests that further refinement of the model or additional data may be required to improve the model's performance in this task.

ACKNOWLEDGEMENTS

The following thesis has greatly benefited from the invaluable insights and direction provided by several individuals, to whom we express our sincere gratitude. Firstly, we would like to extend our heartfelt appreciation to our Thesis Advisor, Dr. Wibool Piyawattanametha, for his unwavering support and guidance throughout the research process. Dr. Wibool's kind-hearted help and encouragement have been instrumental in our success, and we are deeply grateful for his contributions.

We would also like to acknowledge the invaluable assistance provided by the members of Dr. Wibool's lab, who helped us in collecting all the necessary data. Their contributions have been crucial to the completion of this research, and we extend our sincere thanks to them for their support.

The authors of this study wish to express their sincerest appreciation to the Department of Biomedical Engineering at King Mongkut's Institute of Technology Ladkrabang for providing them with the invaluable opportunity to undertake this research endeavor. The authors of this manuscript wish to express deep appreciation for the invaluable aid and backing provided by our esteemed benefactors, whose contribution has proven pivotal in fostering our scholarly and vocational growth.

Ultimately, we wish to convey our profound appreciation to our parents, professors, kinsfolk, and acquaintances for their unwavering assistance and motivation. The steadfast conviction held by our supporters, coupled with their ceaseless reinforcement and provision of assistance, has proven pivotal in facilitating the attainment of our objectives. The progress that has been attained hitherto would not have come to fruition without the invaluable contribution of every individual therein, and we express profound appreciation for their affection and endorsement.

Anon Suphatphon

Pattapon Tanankakorn

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Figures Page

Figures Page

CHAPTER 1

INTRODUCTION

The COVID-19 pandemic, which began in December 2019, has become one of the most devastating pandemics of our time. The causative agent of this pandemic, SARS-CoV-2, is a highly infectious virus that can cause respiratory infections ranging from mild to severe. In this context, early detection of COVID-19 infection is crucial in controlling its spread and preventing severe illness. [1]

Wearable gadgets have emerged as a promising tool for tracking the progression of infectious diseases by continuously monitoring vital signs. We believe that data collected from smartwatches can be used for pre-symptomatic detection of COVID-19, enabling early intervention and treatment. Moreover, the current COVID-19 pandemic has led to overcrowding in hospitals, making it challenging to manage patient care effectively. Therefore, enhancing effective self-isolation and early treatment through remote patient monitoring is critical in preventing the transmission of the disease.

In addition, millions of people worldwide use smartwatches and other wearable devices to monitor various physiological parameters, such as heart rate (HR), blood oxygen saturation (SpO2), blood pressure (BP), and temperature. Real-time monitoring of these parameters has been aided by wireless body area networks (WBAN), which can provide patient monitoring systems to collect ECG, SpO2, temperature, and other bodily characteristics for hospital monitoring, long-distance diagnosis, and other purposes. [2]

Therefore, in this bachelor's project, we aim to develop a pre-symptomatic detection system for COVID-19 based on deep learning technology using data collected from smartwatches. We believe that this system will be an important tool for early detection and management of COVID-19 infections, enabling timely intervention and preventing the spread of the disease.

1.1 Background and significance of the study

Numerous people have been affected as a result of the Covid-19 pandemic, which has also severely burdened healthcare systems around the world. In order to stop the transmission of this virus, it is essential to identify and isolate sick people as soon as possible. The existing testing techniques, however, are pricey, intrusive, and necessitate the use of specialized tools and qualified workers. Alternative methods of Covid-19 detection are therefore urgently required. Wearable technology has grown in popularity in recent years, and many people now use it to monitor their vital signs. Therefore, using the information from these devices to detect Covid-19 could be a promising strategy.[3]

The patient monitoring system consists of three layers consists of data acquisition, storage, and deep learning base. The data-acquisition layer is the layer where the sensor is important. The patient monitoring system uses the sensor to collect information about the patient's vital signs such as sleeping rate, temperature, heart rate, SpO2, etc., and send this data to cloud storage.[4] We use wearable gadgets for real-time monitoring on the website. Wearable gadgets are classified into primary nodes, the node for vital sensor signs, allowing the wireless network to collect information about patient vital signs, symptoms, treatment plans, and communication through the system based on wireless sensors. Storage layer or cloud computing is a method to process the information on the cloud to increase the speed and efficiency of the treatment system, help to analyze all patient data, including demographics, symptoms, and treatment plan, and make all information available to support doctor decisions. In the deep learning base layer, our deep learning will receive the data from patients, and doctors can monitor and receive the alerts from our deep learning, then can treat or help patients in enough time.[5]

Machine learning and deep learning are systems that can learn and solve problems by data patterns in the database, allowing the system to familiarize itself with the existing pattern or data algorithm to develop the system and improve the solution. Deep learning is a particular kind of machine learning that achieves great power and flexibility by learning to represent the world as a nested hierarchy of concepts and representations, with each concept defined in relation to simpler concepts and more abstract representations computed in terms of less abstract ones. The main purpose of this system is to analyze vital sign data of patients by using a supervised learning model including SVM, KNN, Naive Bayes, J48 classifier, and neural network.[6] Support vector machine (SVM) is an algorithm that can help classify information by using data analysis and classifying data through the principle of coefficient determination of equations to create dividing lines for data groups that are sent into a deep learning system.[7] This model is used to solve data grouping problems (classification) and analysis of regression (regression), which have a similar pattern to (logistic regression).

1.2 Objectives

1.2.1: The primary objective of this bachelor's thesis is to enhance the safety of medical staff and physicians by establishing a centralized system that links vital signs data from Covid-19 patients to their beds.

1.2.2: Another goal of this thesis is to utilize deep learning, machine learning, and AI to enable disease identification through vital signs analysis, allowing for the detection of Covid-19 using only smartwatches.

1.2.3: Additionally, this research aims to advance telehealth in Thailand to address the shortage of medical equipment and staff in hospitals, which will aid in reducing the burden on healthcare providers.

1.2.4: By leveraging smartwatch technology, this thesis seeks to develop a Covid-19 detection system that eliminates the need for expensive and invasive testing methods like ATK and PCR, thereby reducing costs and potential risks.[8]

1.2.5 Finally, this thesis aims to design and develop a pre-symptomatic detection system for pandemics and other physical diseases, which will utilize vital signs data from smartwatches. This system could significantly improve disease management and help prevent pandemics like Covid-19 from spreading.

1.3 Scope of the study

1.3.1 Focusing on 2 brands of smartwatches including Fitbit and J-style.

1.3.2 Alert pre-symptomatic system especially Covid 19 and some kind of influenza.

1.3.3 Acquired data from volunteer and BME KMITL staff.

1.3.4 Utilizing vital signs such as heart rate, steps, sleep, and SpO2 to analyze disease pre-symptomatic systems.

1.4 Report Outline

The rest of this report is organized as follows:

Studying the essential and necessary concept of creating the application and retrieving data from the smartwatch by connecting to smartwatches, using Python code for retrieving data from the Fitbit platform and developing our own alert system to be artificial intelligence for detecting Covid-19. Training the anomaly and normal data to observe the difference between the two data to understand the algorithm's working better. Also, improving the detection of pre-symptomatic of Covid-19 systems. The content within this report is organized as follows:

Chapter 2 reviews content over the essential statement and information related to application development, telemedicine website, creating the token and API on the Fitbit platform, and training the AI technology.

Chapter 3 design the metrology of our project, using Python code for retrieving the data of the Fitbit platform and building AI technology, using Dart language for developing our application like cross-platform application in both Android and iOS and implementation of the method conducted to achieve the data and result for the experimenting discussion.

Chapter 4 demonstrates the following data from the practical methodology set for the experimenting process.

Chapter 5 summarizes the conclusions as the final session within this report, reviewing the task achievement and defines future work that is discussed with a particular focus on the discussion according to the experimental result.

CHAPTER 2

REVIEW OF THEORY RELATED

This chapter is about the theory that is related to our project that is considered during the analysis and design phase of this project. The investigation served no purposes: firstly, we wished to identify from telehealth (section 2.1); and secondly, we wished to establish Covid-19 and vital signs (section 2.2-2.4). Finally, (section 2.5-2.12) are about deep learning, smartwatches, programming, and summaries of the chapter.

2.1 The telehealth monitoring system

With the aid of telehealth technology, patients can be monitored for their clinical signs in the comfort of their own homes or anywhere. Vital signs like blood pressure, oxygen saturation levels, glucose levels, peak flow rates, and weight can be tracked using equipment that is installed in the patient's house. The patient uploads these measurements to a secure internet server. The patient's vital signs will have parameters specified that indicate the range of readings that are suitable for that particular patient.[4]

Using a color-coded alert system, the patient's uploaded data is marked. By regularly connecting onto the secure internet server on a computer, the Specialist Nursing Team may monitor the patient's health status and spot any slight changes in their condition.[9]

Researchers measured heart rate and respiration rate by extracting periodic signals induced by chest motion using phase differences in channel state information (CSI) acquired with Wi-Fi signals. did. The system consists of a standard network interface card (NIC) and uses a smartphone/computer as an access point. Heart rate was detected using a Fast Fourier Transform (FFT) based method and respiratory rate was estimated in a single subject using a peak detection algorithm and using the Root MSC (Multiple Signal Classification) method. Estimated on multiple subjects. The system he tested in three different environments.[10] A room crowded with desks and PCs, a walkthrough of her scenario where the subject left the recipient's room, and a long corridor. The authors report achieving a mean heart rate error of 1 bpm and a mean respiratory rate error of 0.25 breaths/min. This CSI-based method is reported to be very robust to respiratory rates sampled at different distances between the transmitter and receiver and in different directions. Unlike his reported UWB radar, the device is cheaper, requires less bandwidth, requires no additional hardware, and can extract data from multiple people in an area, but errors become larger. However, the measured cardiac signal is much weaker than the respiratory signal, and diastole and systole produce only minor fluctuations in the reflected signal, making them more difficult to detect accurately. Additionally, the mean error (MAE) increases during over-the-wall monitoring and at long distances.[11]

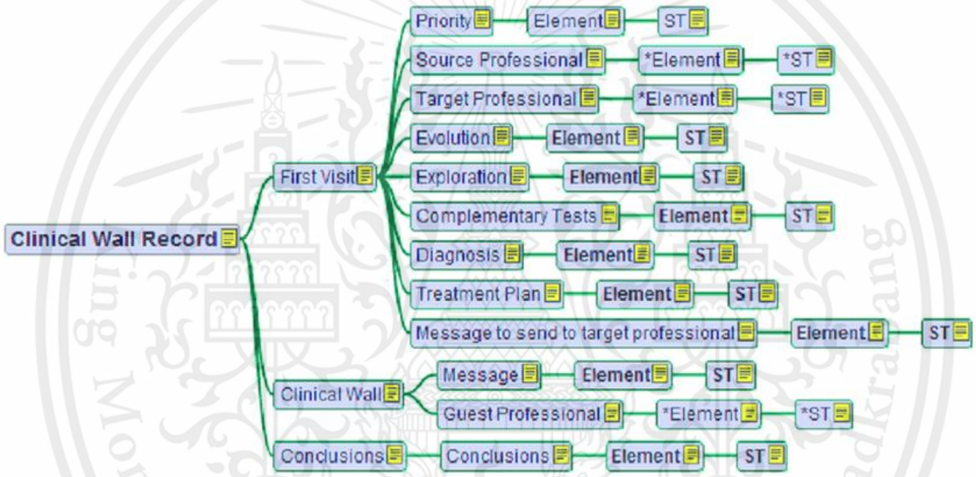


Figure 1: Telehealth clinical wall mindmap. [4]

2.2 Vital Sign Theory

The term "vital signs" refers to the most important sign of living beings that indicate whether the body is normal or abnormal. They consist of four physical signs (physical signs detectable by doctors): temperature, pulse, breathing, and blood pressure. Vital signs are indicators of a person's health. Vital signs are used to determine a patient's health status. Vital signs changes can indicate changes in bodily functions. Vital signs in our project include Heart rate (HR), Steps rate (S), Sleep cycles, Skin temperature (T), and Oxygen saturation percentage (SpO2). [2]

2.2.1 Heart rate (HR)

The heart rate is the number of times of the heart beats in the unit beat per minute. The normal heart rate for men is 72 beats per minute, while the rate for women is 75 to 80 beats per minute. Heart rate can change depending on the physiology of the body, such as the body's oxygen demand and carbon dioxide excretion. Things that affect heart rate include bodily activities such as exercise, sleep, illness, sleep cycles gestion, and certain medications. If the heart beats irregularly, it is called arrhythmia. Sometimes it may be a sign of disease. To measure heart rate by smartwatches, will mostly use a photoplethysmography method to measure heart rate signs.

Photoplethysmography (PPG) is a simple optical technique used to detect blood volume changes in the peripheral circulation. This is an inexpensive, non-invasive method of taking measurements on the surface of the skin.
This technology provides valuable information about the cardiovascular system. Recent technological advances have revived interest in this technique, and it is now widely used to measure and monitor clinical physiology.[12]

Fundamentally, photoplethysmography works on a simple premise. Blood is red because it reflects red light and absorbs green light. By combining a green LED with a photodiode, smartwatches use its own Pure Pulse technology to detect the amount of blood flowing down your watch. When the heart is beating, this flow, and therefore the amount of green light absorbed, is greater. These lights blink hundreds of times per second to get the most accurate BPM (beats per minute) data.
All companies dabbling in this optical sensing technology follow the same pattern, but the accuracy of the readings, of course, depends on how each company's algorithms interpret the data.

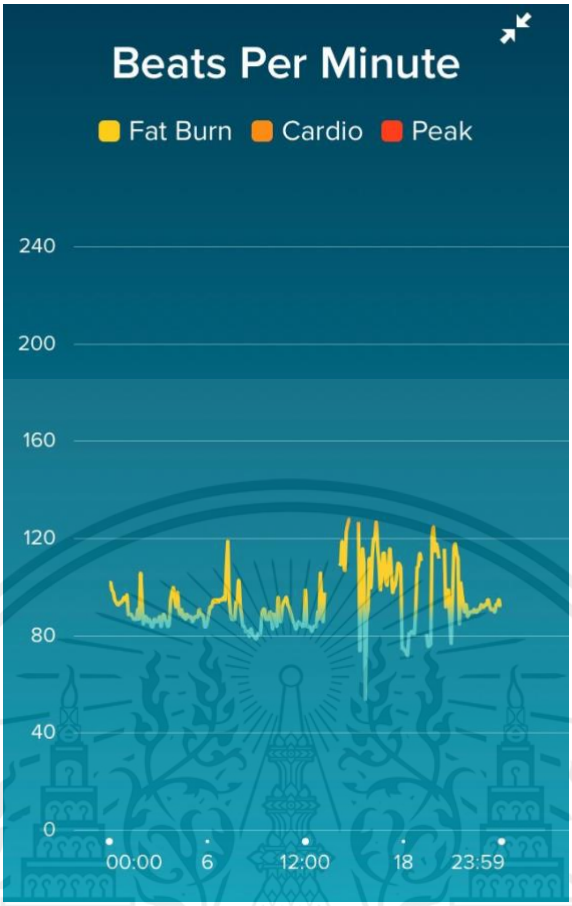


Figure 2: Example of heart rate cycle from Fitbit mobile app.

2.2.2 Steps counting (S)

Health experts have long touted the benefits of moderate exercise for at least two to three hours a week. For most adults, the baseline for that level of aerobic activity is walking at a pace of 3 miles per hour or more. Another commonly used metric is the goal of 10,000 steps per day.[13]

Smartwatches have a 3-axis accelerometer. Then when we are moving back and forth, left, and right, up and down, they can track our movements. And by processing the recorded movement data, Smartwatches can determine if you're walking (or running) or just bumping your hand against the desk. Most smartwatches use complex algorithms to determine when you're walking on your own and when, say, you're in a car or moving your arms but not moving at all. The step count at the end of the day is very accurate because they filter out movements that we don't consider to be steps.[14]

Figure 3: Example of steps counting from Fitbit mobile app.

2.2.3 Sleep cycles

Sleep cycles is about tracking how long you slept and how much time you spent in each sleep stage. These smartwatches use your movement and heart rate information to estimate how long you spend in light, deep, and Rapid Eye Movement (REM) sleep each night. Smartwatches only know you're sleeping when your body has been resting for an hour without any significant movement. After your body has passed that time, smartwatches will start collecting data about your sleep quality.[15]

For the first few hours (less than three hours), the smartwatches sort sleep into her three categories: restless, awake, or asleep. Once you cross the 3-hour threshold, it starts tracking different types of sleep called sleep stages. These stages include bright stage, deep stage and REM stage.[16]

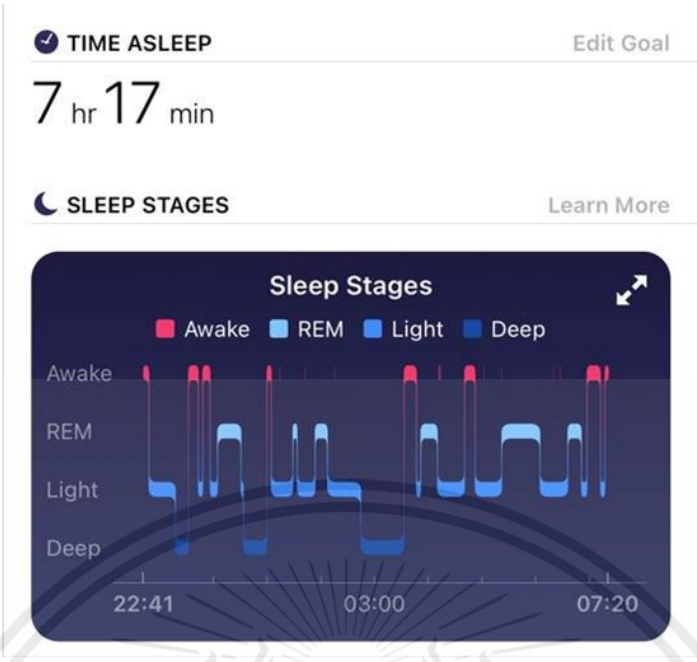


Figure 4: Example of sleep cycle from Fitbit mobile app.

2.2.4 Skin temperature (T)

Smartwatches' temperature feature monitor as skin temperature. For the measurements are taken at night while you sleep in Fitbit watches and are taken immediately at the moment for Jstyle watches. Fitbit can then view nightly averages and trends for the last seven days on the Health Metrics dashboard.[17]

Even two brands have different time to show the data but the value that we can measure from each watch is not much different to each other, show like that the sensor or the program that measure skin temperature is very likely or can be the same.

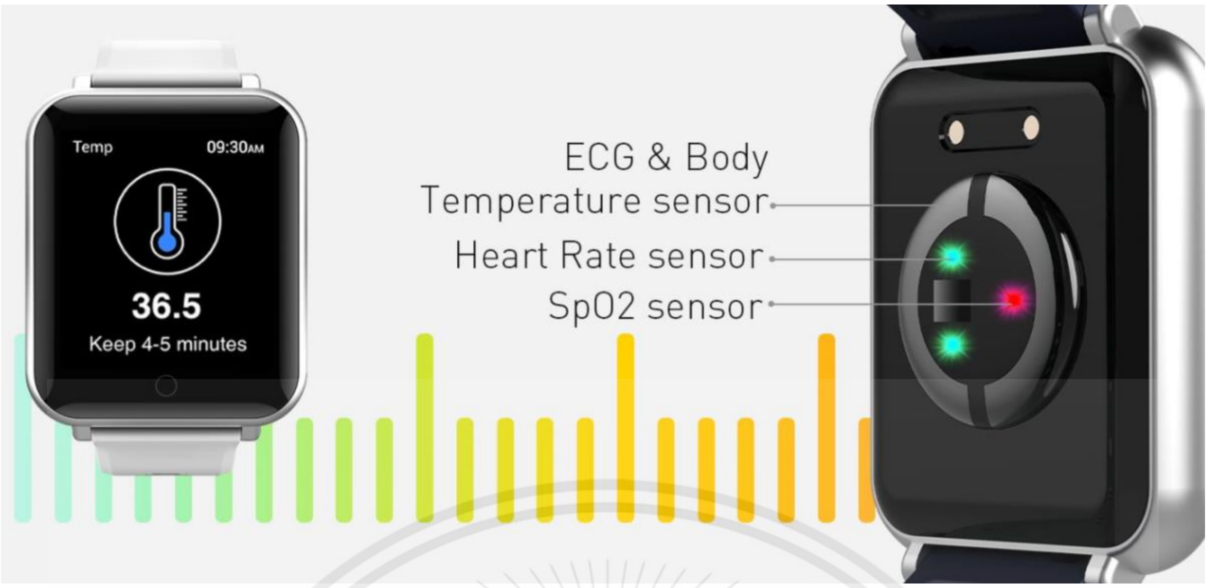


Figure 5: Example of skin temperature from J-watch smartwatch.

2.2.5 Oxygen saturation percentage (SpO2)

Oxygen saturation refers to the percentage of hemoglobin in oxygen-rich arterial blood. This is a critical parameter for the Respiratory Circulation System. Many respiratory diseases cause oxygen saturation to be lower than normal. The following factors may contribute to a decrease in oxygen saturation.[18]

For Jstyle, it can measure SpO2 value immediately by using another hand's index finger to hold the watch to be stable for measure.

For Fitbit, it aims to show users an average of their blood oxygen saturation measured during their last sleep. This information is displayed to the user on their wrist-worn Fitbit device or connected mobile device. Fitbit SpO2 is a standalone general wellness software product that uses data from a consumer.

Smartwatches with an optical sensor (PPG). PPG uses red and infrared LEDs to illuminate the skin. The reflected light is measured, and PPG data is provided. These sensors are commonly used for general wellness purposes, such as heart rate measurement, sleep monitoring, and other general health features found on fitness smartwatches.

Figure 6: Example of SpO2 from JClife application.

2.3 Covid-19

Covid-19 is an infectious disease caused by the most recently discovered coronavirus, the SARS-CoV-2 virus, and it is a virus that can infect both humans and animals. The virus and emerging infection were unknown before the outbreak in Wuhan, China in December 2019. Currently Covid-19 There is a widespread epidemic affecting many countries around the world. [19]

2.3.1 Symptoms of Covid-19 infection

This disease includes respiratory symptoms that will affect the changes in the vital signs of the infected person.

2.3.1.1 Common symptoms of Covid-19

- Fever
- Cough
- Tiredness
- Loss of taste or smell [20]

2.3.1.2 Serious symptoms of Covid-19

• Difficulty breathing or shortness of breath
• Loss of speech or mobility, or confusion
• Chest pain

In more severe cases, Covid-19 infection can cause pneumonia, severe acute respiratory syndrome and sometimes death. [21]

2.3.1.3 Variants of Concern

There are two types of Variants of Concern [22]:

• Alpha Variant (B.1.1.7): This variant has been found to be at least 50% more contagious than the original COVID-19 virus. Globally, the R0 for COVID-19 varies, but with the Alpha variant, the R0 is increased at least 50%, leading to an exponential increase in transmission.

• Delta Variant (B.1.617): This variant is highly contagious, spreads quickly and is immune-evasive.

• Omicron (B.1.1.529): Omicron is the newest variant in the current outbreak. This variant is caused by mutations in the coronavirus genes in more than 50 locations, 32 of which occur on spike proteins also known as Spike Protein, which is a protein that viruses use to enter the human body's cells and is found in more than all species and twice as much as the delta species. Moreover, Omicron is found to be a mutation in the receptor, and it attaches to human cells in more than 10 locations. The virulence of the Omicron variant tended to be lower compared to Delta variant. However, the infection can cause severe symptoms and damage the lungs in people who have not been vaccinated or in people with diabetes, heart disease, young children, Etc., and can spread quickly.

• Epsilon (B.1.427 and B.1.429): The specific mutations in the Epsilon variant including three in the spike protein help the virus to dodge the protection given by the COVID vaccines or past Covid-19 infection. After testing the resilience of blood plasma from vaccinated people and those who had already had COVID, researchers found that the ability to neutralize the Epsilon variant was reduced to 2 and 3.5 times.

2.4 Interpreting Diagnostic Tests for SARS-CoV-2

The coronavirus virus pandemic of 2019 continues to devastate most of the world. Diagnostic tests for the severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2) are currently being developed, and a clear understanding of the nature of the tests and how to interpret their results is crucial. This Viewpoint discusses how to interpret two methods of SARS-CoV-2 diagnostic testing—reverse transcriptase-polymerase chain reaction (RT-PCR) and enzyme-linked immunosorbent assay (ELISA) and how the results may alter over time. [23]

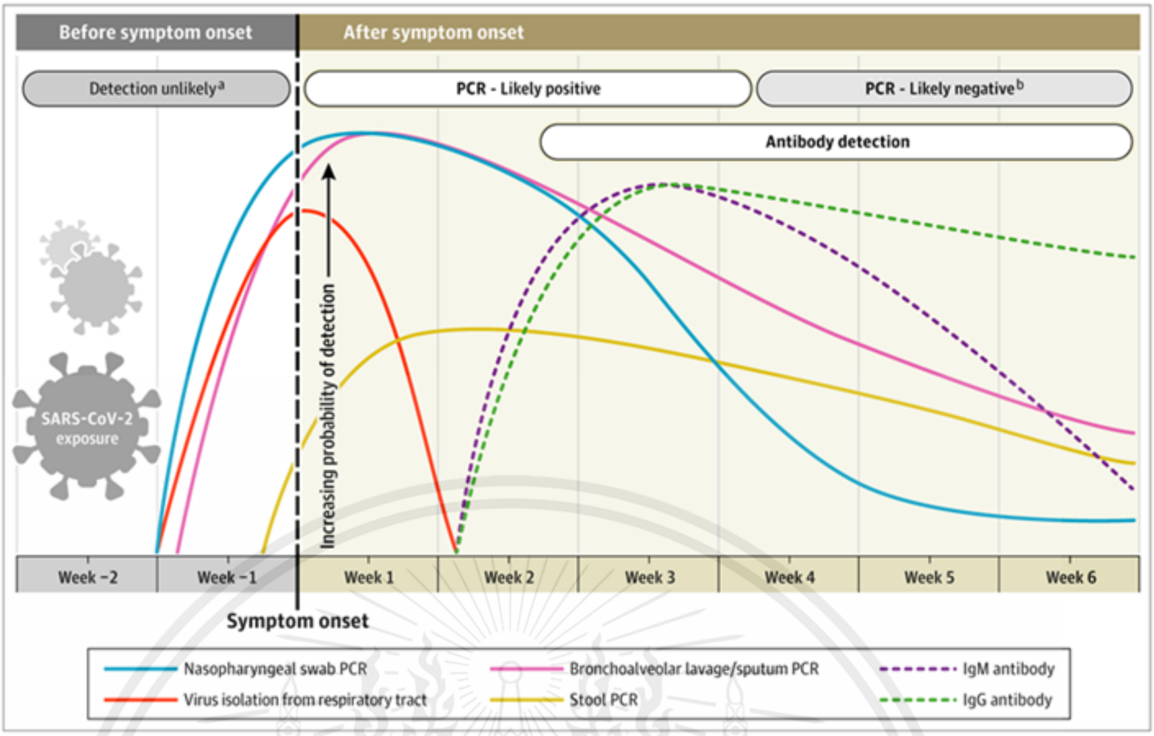


Figure 7: Estimated Time Variation in Diagnostic Tests for SARS-CoV-2 Infection Detection Relative to Symptom Onset

Data from numerous published papers was used to estimate time intervals and virus detection rates. Because values vary between research, estimated time periods should be regarded as approximations, and the chance of detecting SARS-CoV-2 infection is reported subjectively. SARS-CoV-2 is an abbreviation for severe acute respiratory syndrome coronavirus 2; PCR stands for polymerase chain reaction.[24]

a.) Detection happens only when patients are actively followed up on from the moment of exposure.
b.) A PCR of a nasopharyngeal swab is more likely to provide a negative result than a positive one.

2.4.1 Detection of Viral RNA by RT-PCR

Too far, the RT-PCR test employing nasopharyngeal swabs or other upper respiratory tract specimens, such as throat swabs or, more recently, saliva, has been the most widely used and trustworthy method for diagnosing Covid-19. Different manufacturers utilize a range of RNA gene targets, with the majority of tests focusing on one or more of the ORF1, RNA-dependent RNA polymerase (RdRp), nucleocapsid (N), spike (S), and envelope (env) genes. With the exception of the RdRp-SARSr (Charité) primer probe, which has a somewhat lower sensitivity possibly due to a mismatch in the reverse primer, the sensitivities of the tests to individual genes are comparable according to comparison studies. However, a “positive” PCR result reflects only the detection of viral RNA and does not necessarily indicate presence of viable virus.[25]

As determined by the cycle threshold (Ct), viral RNA in the nasopharyngeal swab in the majority of patients with symptomatic Covid-19 infection becomes detectable as early as day 1 of symptoms and peaks during the first week of symptom onset. A fluorescent signal is produced by a certain number of replication cycles, or Cts, and lower Ct values indicate larger viral RNA levels. A Ct value less than 40 is considered clinically positive for PCR. By week three, this cheerfulness begins to wane and eventually disappears. However, the Ct values found in hospitalized patients with severe illnesses are lower than those found in mild cases, and PCR positive may last longer than 3 weeks following the commencement of the disease when the majority of mild cases would provide a negative result.

2.4.2 Detection of Antibodies to SARS-CoV-2

The pandemic response still requires the accurate and sensitive detection of SARS-CoV-2 antibodies. Laboratory-based confirmatory and reference activity must be measured in order to predict neutralizing activity and the immune response.

The primary target antigen for neutralizing antibodies is the viral receptor binding domain (RBD). In a novel hybrid approach, a solid-phase S1 preferentially presenting RBD was combined with a labeled RBD conjugate and used in a two-step sequential assay for the detection and measurement of antibody to RBD, taking advantage of the double antigen binding assay (DABA), which offers the most sensitive format (anti-RBD). The SARS-CoV-2 hybrid DABA uses both S1 and RBD viral antigens in a two-step double antigen binding format. The solid phase is covered with S1 antigen, while horseradish peroxidase (HRP)-RBD conjugates show captured antibodies.[26]

2.5 Data analytics

Data analytics is the method of gleaning insights from data that's extricated, changed, and centralized to find and analyze covered up designs, connections, patterns, relationships, and peculiarities, or to validate a hypothesis or theory.

Within the past, information was analyzed to create future choices. Nowadays, information can be analyzed to create real-time choices, spot developing patterns and reveal bits of knowledge that would not be apparent utilizing bequest information forms.[27]

Information examination can offer assistance to move forward commerce forms. The information can give a clearer picture of what's productive and what's not, and analysts can penetrate more profoundly into the information to find root causes.

Information investigation boosts income by permitting individuals to form speedier and more educated choices. With enough data to analyze, businesses can foresee customers' behavior, get their needs, and react in real-time by changing or including items to meet the shown request. This may result in a competitive advantage, made strides client encounters, and moved forward procurement and maintenance of clients.

There are basically four diverse sorts of information analytics:

- Expressive analytics: This sort of analytics must do with what happened with analyzed information over an indicated period of time.
- Demonstrative analytics: Symptomatic data analytics appears as the "why" in an information slant. This includes having a more profound see into why certain designs were shown within the information.
- Prescient analytics: The objective here is to predict what is anticipated to happen within the future based on the results of analyzed information over time.
- Prescriptive analytics: In prescriptive analytics, information investigation is utilized to form proposals on what to do another.

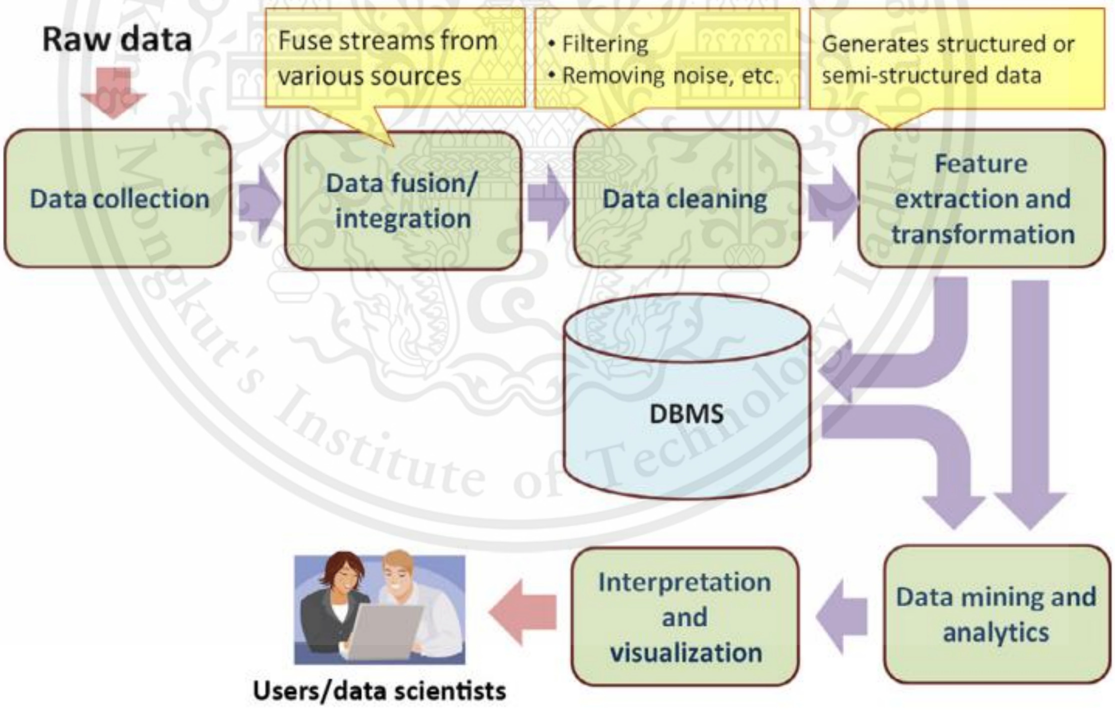


Figure 8: Data analytics in datacenters [27]

When data is input, it has to be changed over and organized to get precise answers from explanatory questions. Distinctive information preparing alternatives exist to do this. The choice of approach depends on the computational and explanatory assets accessible for information handling.[27]

Data fusion is the process of combining data from multiple sources to create a more complete and accurate representation of a particular phenomenon. Benefits of data fusion include improved accuracy, increased reliability, and improved situational awareness. However, data fusion also has the following limitations: B. The potential for data overload and the need for expert analysis to identify and resolve discrepancies between different sources. Overall, data fusion is a valuable tool for improving decision making in various applications such as intelligence analysis, remote sensing, and environmental monitoring.

Data cleansing involves scrubbing for any errors such as duplications, inconsistencies, redundancies, or wrong formats. It's also used to filter out any unwanted data for analytics.

Feature extraction and transformation are important processes in data processing that involve creating new variables or modifying existing variables to improve the accuracy and performance of machine learning models.

Data mining is the process of discovering patterns, trends, and information from large data sets. It involves analyzing and extracting useful information from large amounts of data, using statistical and computational techniques. It is an essential component of data processing as it helps organizations make more informed decisions by identifying hidden patterns and relationships in their data. It can be used in many different fields including marketing, finance, healthcare and scientific research.

Interpretation and visualization are crucial constituents in the process of data processing. After the processing of data, it is essential to present the derived information in a comprehensible and actionable manner.
The process of interpretation pertains to comprehending and scrutinizing the manipulated data in order to ascertain patterns, associations, and inclinations. Fluent comprehension of statistical and analytical methodologies, coupled with informed expertise in a particular field, is an essential prerequisite to effectively decipher the outcomes. The knowledge acquired through interpretation may subsequently serve as a valuable resource towards enhancing organizational outcomes and informing decision-making.
The process of visualization entails the depiction of data that has been subjected to processing procedures, in a manner that is conducive to comprehensibility and aesthetically pleasing to the viewer. The aforementioned encompasses various types of visual aids, such as charts, graphs, maps, and other forms of graphical depictions utilized to represent data. The implementation of visualization techniques can aid in the recognition of obscured patterns and correlations that may not be readily discernible through the examination of unprocessed data alone. Additionally, these visual representations facilitate the communication of perceptive findings to diverse audiences.[27]

2.6 Machine learning and deep learning

It is a self-learning and problem-solving system by recognizing patterns in the data from the database. It allows the system to recognize the basic patterns of algorithms and datasets. available to develop appropriate problem-solving processes. The main purpose of this system is to simultaneously monitor the health status of patients remotely. It is alerted if the resulting data has abnormal value and provides analysis using supervised learning models such as SVM, KNN, ellipse, isolated forest, local peripheral model (LOF) and neural networks. When the model determines that the patient is in critical condition, the system will issue an alert. In remote patient monitoring systems, this is achieved through the development of AI and data collection of vital signs such as respiratory rate (RR), heart rate (HR), Oxygen saturation percentage (SpO2) and blood pressure (BP).[28]

2.6.1 StandardScaler

StandardScaler is an important tool used primarily as a preprocessing step before many machine learning models to normalize the input dataset's functional range.

When the range of the features in the input dataset exhibits significant variance or when the features are measured using different units of measurement, StandardScaler must be used.

By subtracting the mean and scaling the data to unit variance, the StandardScaler approach is used to center the data. But even so, the presence of outliers can clearly affect how the empirical mean and standard deviation are calculated, so limiting the range of characteristic values.

The dissimilarities exhibited in the initial characteristics of the data may pose challenges for various machine learning models. As an illustration, for models predicated on distance computation, the prevailing characteristic in the event that a feature exhibits a significant range of values shall be the distinctive attribute governing the distance.

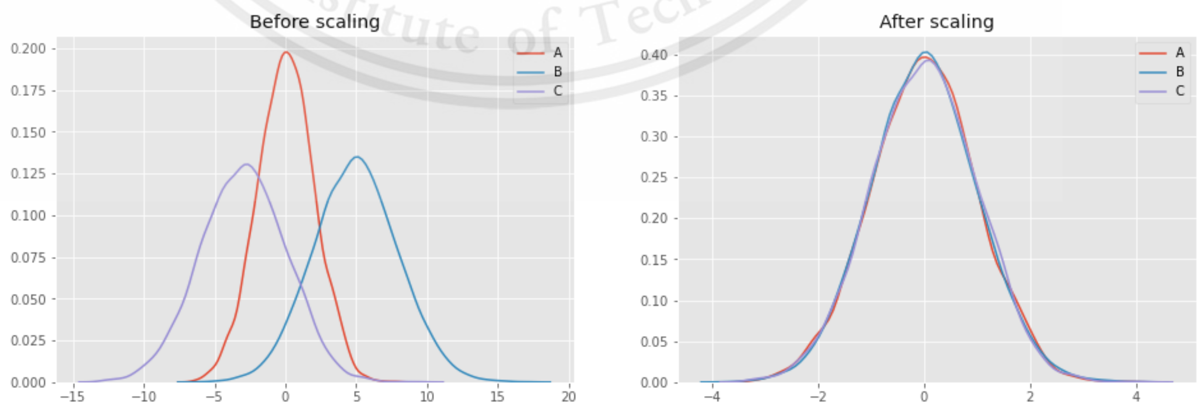


Figure 9: The graph of data before and after using StandardScaler. [29]

The StandardScaler methodology proposes that variables assessed at diverse scales exhibit unequal influence on the model fitting and learning mechanisms, and thus possess the potential to instill bias.

In order to mitigate the potential issue at hand, it is imperative that the data being utilized for integration into the machine learning model must undergo standardization. This involves the normalization of the data, typically with respect to a mean of zero and a standard deviation of one ($\mu = 0, \sigma = 1$).

2.6.2 Isolation Forest

The isolated forest uses the random forest technique by randomly pointing at information and generating a tree that is the data branch. When searching, normal information will have trees with the same number if we find information with fewer trees than other trees. The isolation forest tries to separate each point in the data. In the 2D case, it generates a random line and tries to distinguish a point. Here, an outlier can be separated in a few steps while closer normal can take many steps to separate. It can be considered as anomalous information, as shown in the figure.

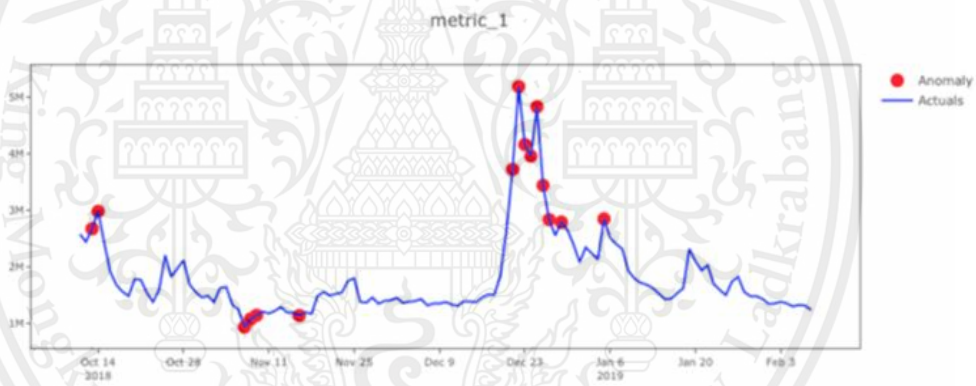


Figure 10: Anomaly information detected from Isolation Forest. [30]

2.6.3 Z-transformation

The z-transformation is a normalization procedure that allows the comparison of points from different distributions. Using the standard deviation and mean of a distribution, the z-transformation combines different distributions with a normalized distribution, allowing for comparison of different figures. The z-score defines the position of the raw data in terms of standard deviations from the mean. If the value is greater than or equal to the mean, the z-score is positive and negative if the value is less than the mean. [30]

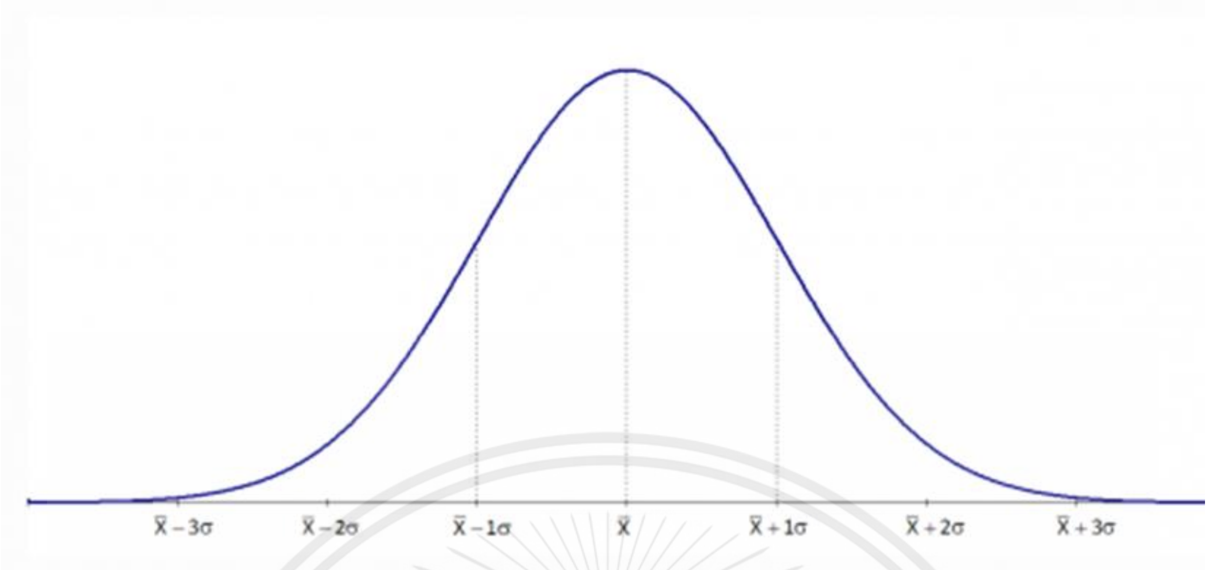


Figure 11: A standard normal distribution

2.6.4 Support Vector Machine

Support Vector Machine (SVM) is a supervised machine learning algorithm that serves as a classification and regression engine. Despite the common characteristics of regression problems, they are particularly suitable for classification tasks. The main goal of the Support Vector Machine (SVM) algorithm is to identify a distinct hyperplane in a multi-dimensional (N-dimensional) space that efficiently separates and classifies discrete data points. The size of the hyperplane depends on the number of existing features. If the number of input features is limited to two, the hyperplane will only show up as a linear structure. When the number of input features is three, the resulting hyperplane transforms into a two-dimensional plane. When the number of traits exceeds three, it becomes difficult to conceptualize the situation. [7]

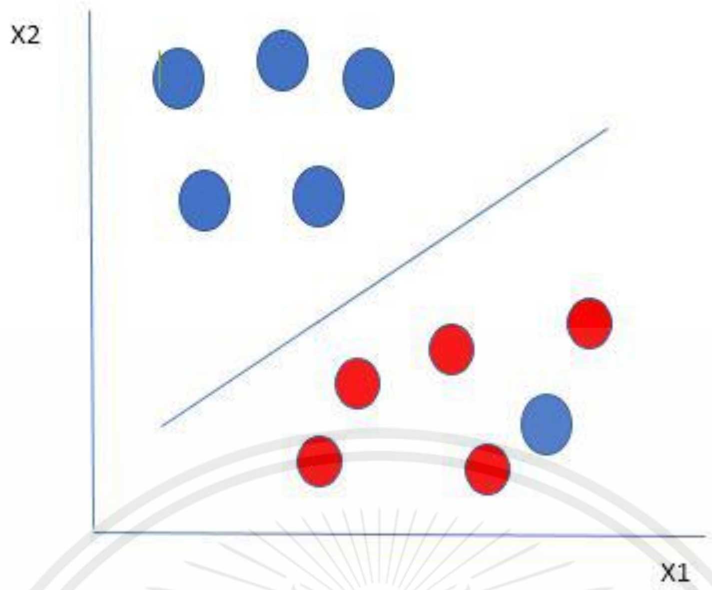


Figure 12: Hyperplane of SVM which is the most optimized one.

SVM finds the maximum amplitude as for previous datasets and adds a penalty every time a point crosses the amplitude. So the profit in these types of cases is called soft profit. When there is a soft margin in the dataset, SVM will try to minimize (1/margin + ∧(∑ penalty)). Losing a hinge is a commonly used penalty. Without breaking, no loss of hinge. In case of breach, hinge loss is proportional to breach distance.

2.6.4 Confusion Matrix

The confusion matrix represents a tabular form utilized to assess the efficacy of a classification model. The aforementioned construct denotes a matrix that comprises of predictions regarding true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). The true classes are denoted by the rows, with the predicted classes being represented by the columns.

		Predicted Class		
		Positive	Negative	
Actual Class	Positive	True Positive (TP)	False Negative (FN) Type II Error	Sensitivity $\frac{TP}{(TP+FN)}$
	Negative	False Positive (FP) Type I Error	True Negative (TN)	Specificity $\frac{TN}{(TN+FP)}$
		Precision $\frac{TP}{(TP+FP)}$	Negative Predictive Value $\frac{TN}{(TN+FN)}$	Accuracy $\frac{TP+TN}{(TP+TN+FP+FN)}$

Figure 13: Type of Confusion Matrix.

Precision is a metric used in the context of a confusion matrix to evaluate the performance of a classification model. It measures the proportion of true positive predictions out of all positive predictions made by the model. In other words, precision represents the accuracy of positive predictions. The formula for precision is
Precision = True Positives / (True Positives + False Positives)

Recall is a sensitivity or true positive ratio, is a metric used in the context of confusion matrices to evaluate the performance of a classification model. It measures the proportion of actual positive samples that are correctly identified by the model as positive.The formula for recall is
Recall = True Positives / (True Positives + False Negatives)

Accuracy is a commonly used measure in classification problems. However, this can be misleading in situations where classes are unbalanced, i.e. one class has much fewer instances than the other. In such cases, other measures such as accuracy, recall, and F1 score may be more suitable for evaluating model performance. The formula for accuracy is
Accuracy = (True Positives + True Negatives) / (True Positives + False positives + True Negatives + False Negatives)

The F1 score is a metric commonly employed to evaluate the accuracy of a given model as it takes into account both precision and recall. The harmonic mean of precision and recall represents a metric that varies between a range of 0 to 1, with the maximum value of 1 connoting faultless precision and recall. The formula for the F1 score is

F1 score = 2 * (Precision * Recall) / (Precision + Recall)

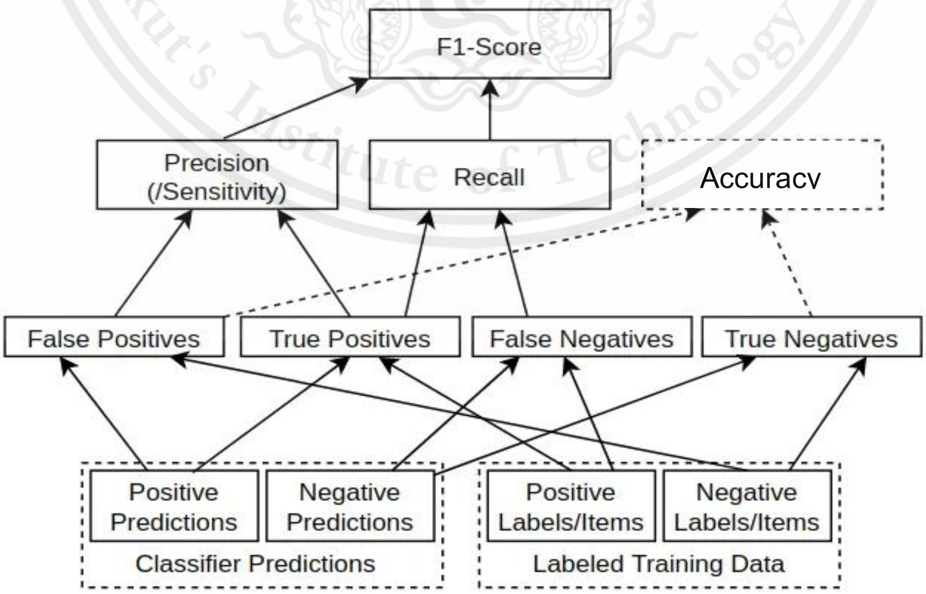


Figure 14: The relationship diagram of components to measure the effectiveness of the model. [32]

2.7 Anomaly detection over Time series

A time series is a series of data points that occur sequentially over time. A time series shows all the variables in a dataset that change over time. Anomaly detection identifies unusual and suspicious events over time. The task of detecting anomalies or identifying outliers entails the process of recognizing atypical or incongruous patterns within a given set of data. Illicit activity frequently leads to anomalous data.

Anomaly detection accounts for sudden spikes and dips in data sets. We can also gain valuable insights from your dataset, but we have to smooth our dataset as a smoothing process. Sudden spikes and dips can lead to inconsistent results during forecasting. Smoothing process of a series with such a moving average has some drawbacks. First, we lose some observations data at the beginning and end of the series. Second, the moving average tends to over smooth the series, making it less responsive to sudden trend changes and jumps.

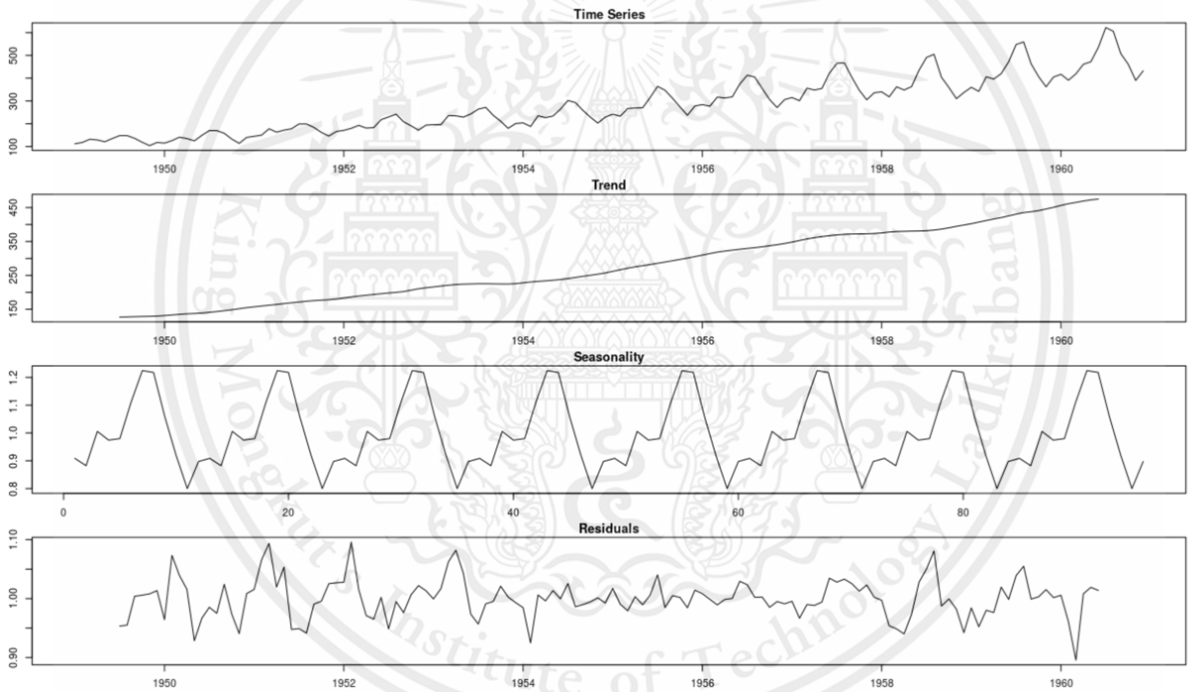


Figure 15: Seasonal decomposition of abnormal detection. [33]

For this project, we are dealing with an additive model. That is, it consists of linear trends and seasonal cycles with the same frequency as width and amplitude as height using four components.

Components:

The trend is the general direction of the series over a long period of time and calculated using the central moving average of the time series. A moving average is calculated with a window length equal to the frequency of the time series. For example, use windows of length 12 for monthly data.

The seasonality is a distinct, repeating pattern observed in regular intervals due to various seasonal factors. Could be monthly, weekly, etc.

The residual is the irregular component consisting of the fluctuations in the time series after removing the previous components.

Noise is undesirable and erroneous data points. Noisy data contains meaningless information and can corrupt time series models in training. Removing noise ensures a high-quality dataset.

2.8 Neural network

The neural network is a branch of artificial intelligence (AI). It is a concept that designs computer networks to mimic the functions of the human brain. Human beings have a learning process by recognizing patterns. They have observed things that are unknown as they see them more often. When humans see it in more different ways, similar things are learned and memorized, and when you see it again, you will be able to tell what you see. The multi-layer perceptron (MLP) neural network structure is coordinated through the node. Aiming on the flexural conduct of prestressed steel-concrete non-stop composite beams, a third-dimensional finite detail numerical evaluation version is established, and the complete method of the check is stimulated primarily based totally on BP neural network. It consists of three layers, which are fully connected to the functions of the nodes, including:

- Input layer, this layer is the part that handles the input data. The number of nodes depends on the number of inputs that data will be introduced into the model.
- Hidden layer, this layer in the middle affects the learning efficiency of the model. This layer can have any number of layers, any number of Neurons. Adding layers and the number of neurons will greatly affect how the model works.
- Output layer, this layer takes the calculated data in the hidden layer to work. The number of nodes exported in this layer depends on the output format to be used as shown in Figure.

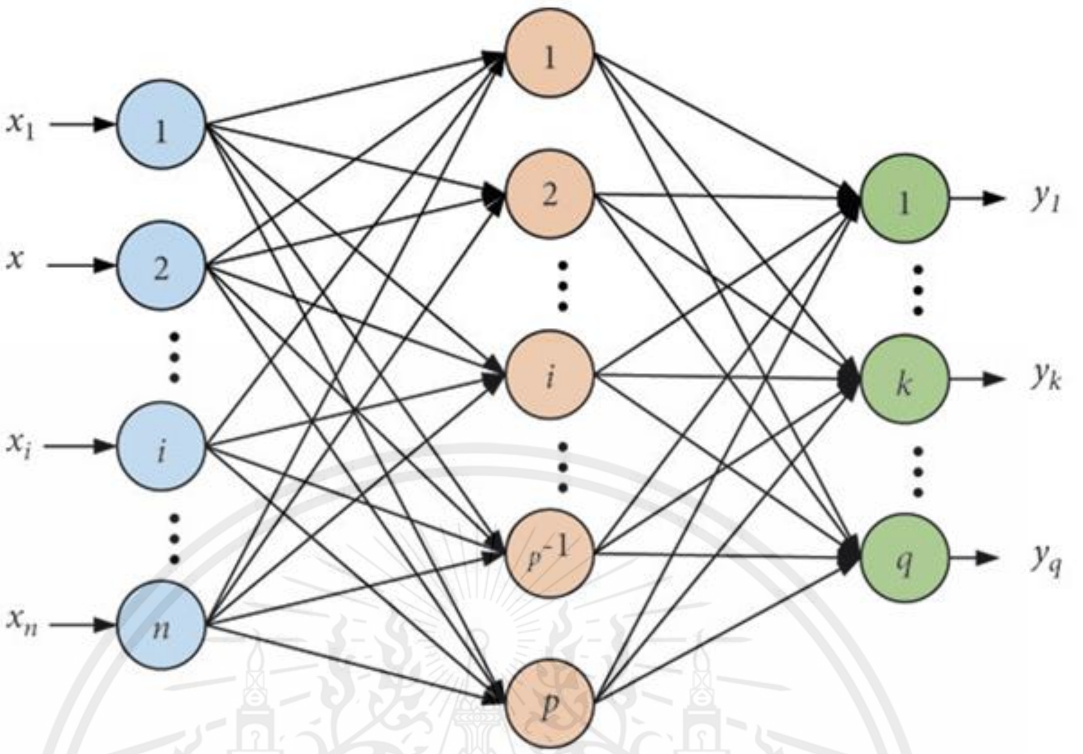


Figure 16: Neural network infrastructure.

2.9 Smart Watch

2.9.1 J-Style 2025E

Figure 17: J-Style 2025E smartwatch. [34]

J-Style 2025E is a smartwatch made in China. J-Style 2025E smartwatch has many functions that include measuring vital signs such as percentage of blood oxygen concentration (%SpO2), blood pressure, and respiratory rate. Moreover, this model can display heart rate and temperature data on the smartwatch dial.

2.9.2 Fitbit Charge 5

Figure 18: Fitbit Charge 5 smartwatch. [35]

Fitbit is one of the world's most popular brands producing smartwatches with beautiful designs, using suitable materials in production, and not easily irritated. The user can be worn all day and use a variety of functions. They can also record our activities and vital signs that can be recorded for 24 hours while doing those activities. The Fitbit Charge 5 is the fitness tracker, packing in everything the company has learned from years of honing its craft. This smart tracker includes options for stress tracking, blood oxygen saturation, heart rate variability (HRV), sleep tracking, and mindfulness sessions to keep yourself mentally ready as well as maintaining your physical fitness.

2.10 Python

Python is a computer programming language of high-level that is interpreted. It is supposed to be an easy-to-read scripting language by obviating the computer program language's structure and syntax complexities. In terms of converting a set of instructions written in machine language to machine language, Python can translate a set of instructions line by line and distribute them to the processor for the computer to function correctly.

Additionally, Python can be used for various programming tasks as it is a general-purpose language.

Figure 19: Logo of Python program. [36]

2.10.1 Usage of Python

- Python is a computer language that is stable, flexible, and simple to understand. It is commonly used for developing websites and software, task automation, data analysis, artificial intelligence, and machine learning.
- Python can be used to create simple games using the programming language and tools that come with the program. It can develop to be used in game industries.
- Python has syntax that can be used to create new programming languages.
- 2.10.1.4 Python can be used in finance for analysis of quantitative and qualitative data to determine asset price trends.
- 2.10.1.5 Python can be used in the field of search engine optimization (SEO) and natural language processing (NLP).

2.10.2 Advantages and Disadvantages of Python

Python is a high computer programming language, which means it can run code line by line. When the error occurs, it will stop the procedure and indicate the error. Python is a dynamic computer programming language that automatically assigns the data typed during execution. The programmer is not required to be concerned about declaration variables. Additionally, Python supports libraries via the Python package management (pip), which simplifies the process of importing additional packages and libraries. This makes Python easier to read, learn, and write and increases productivity by removing the need for programmers to spend excessive time understanding the programming language's syntax. However, the line-by-line execution function, including the response from dynamically typed code, is frequently delayed since it must perform additional work when running the code, resulting in future runtime errors.

2.10.3 Python Library

Python's standard library is quite large, as demonstrated by the extensive contents page listed below. The library includes built-in modules written in C language (C/C++) that enable access to system capabilities such as file I/O that would otherwise be unavailable to Python programmers, as well as Python modules that provide standardized solutions to a variety of common computer programming problems.

- System-specific parameter

System-specific parameter and function is the module gives access to certain variables that the interpreter uses or maintains and functions that have a strong interaction with the interpreter.

- Argparse

The argparse package simplifies the process of developing graphical command-line interfaces. The program specifies the arguments required, and argparse.

determines how to extract them from sys.argv. Additionally, the argparse module creates help and usage instructions and generates errors when users supply invalid arguments to the program.

- Pandas

This is used for data analysis. Pandas is based on two essential Python libraries, which are NumPy for mathematical computations and matplotlib for data visualization. Pandas act as a layer around these libraries, letting you quickly access various matplotlib and NumPy methods. Allowing users to plot a graph within a few lines.

- Numpy

NumPy is one of the Python libraries for array manipulation. Additionally, it has functions for working with linear algebra, the Fourier transform, and matrices. It has a slew of helper functions that make working with arrays fast.

- Matplotlib.pyplot

It is a set of functions that simulate the behavior of MATLAB in matplotlib. Each pyplot function modifies a figure in some way by creating a figure, defining a plotting region within a figure, charting certain lines within a plotting area, or labeling the plot.

- Matplotlib.dates

Matplotlib provides extensive data plotting capabilities, building on the roots of Python's DateTime module, as well as incorporating modules pytz and dateutils.datetime data are changed to a numeric number that reflects the number of days since 0001-01-01 UTC.

- Seaborn

Seaborn is a Python module for creating statistical visuals. It is based on matplotlib and tightly integrated with Pandas data types. Seaborn aids in the exploration and comprehension of user data. Its charting functions operate on dataframes and arrays that include entire datasets and do the required semantic mapping and statistical aggregation internally to generate relevant graphs.

- Statsmodels

Python package includes classes and functions for estimating various statistical models, executing statistical tests, and exploring statistical data. Statsmodels are based on NumPy, SciPy, and matplotlib, but they include more advanced statistical testing and modeling capabilities.

- Sklean

It is a Python package that implements various unsupervised and supervised learning techniques. It is the most valuable and comprehensive machine learning library available. It provides a set of efficient machine learning and statistical modeling methods, such as classification, regression, clustering, and dimensionality reduction, via a Python-based consistency interface. This Python-based library is based on NumPy, SciPy, and Matplotlib.

- Easydict

EasyDict enables the use of dict values as variables work repeatedly. A property dot notation similar to JavaScript for Python dicts.

2.11 Dart

In order to give developers, access to an object-oriented language with static type analysis, Google created the open-source language Dart. Dart has seen significant changes since its 2011 debut, both in terms of the language itself and its main objectives. Unlike many other languages, Dart was created with the intention of facilitating and speeding up the development process for developers. It therefore comes with a sizable collection of integrated tools, including a parser, formatter, and its own package manager. Additionally, the Just-in-Time build, and Dart virtual machine enable code modifications to be immediately executed. No extra environment is needed for the code to run once it is in production because it may be compiled in native language. For web development, Dart is transpired to JavaScript.

Figure 20: Logo of Dart language. [36]

In terms of syntax, Dart is very similar to languages like JavaScript, Java, and C++, so if you know any of these languages, learning Dart will take hours.
Additionally, Dart has strong support for asynchronicity, and working with generators and iterables is very easy.

2.12 Flutter application

A Strong and Reliable SDK for Cross-Platform Mobile App Development is Flutter. Google developed Flutter, a cross-platform software development kit (SDK) for building mobile applications. To create apps for Android and iOS devices, Flutter employs the Dart programming language. A single code base may be used to create apps with a native look and feel on both Android and iOS devices because it is cross-platform. Flutter is strong and dependable, providing limitless creative opportunities to swiftly create stunning applications. The development cycle on Flutter is swift due to its capabilities and architectural choices, making it appropriate for creating both complicated apps and games as well as speedy MVPs and prototypes. It is important to create apps that work on different screen sizes because it allows more people to use your app. Not everyone has the same screen size, so it's important to make sure your app can be used by as many people as possible.

Additionally, building an app that works across multiple screen sizes can make it easier to reach a global audience, as people in different regions are likely to have different sizes and specs. Different screen techniques. Creating an app that works on any screen size is key to maximizing its usability and appeal.

Figure 21: Logo of Flutter application. [37]

Google launched Flutter in 2017 as an open-source framework for developing cross-platform applications. The goal of the project is to give developers a way to build high-quality, natively compiled apps for Android and iOS from a single codebase. One of the major limitations of creating native iOS or Android apps is the time and effort required to create two separate code bases. Building a cross-platform application with a framework like Flutter can help reduce development time and costs and make it easier for users around the world. Additionally, building apps that look original improves user experience and increases adoption.

As mobile developers, we are often asked whether to use a cross-platform solution or create native apps. While we always have a considered answer, we almost always turn to a budget. It's important to note that the cost of running the same application on separate code bases is better spent on better funded projects, where native performance can really be the defining feature. determined. User experience. Flutter uses the Dart programming language, a modern object-oriented language created by Google. With native and web development support and the use of powerful tools like the Flutter SDK, App Engine, and Firebase, Dart makes it easy to build high-quality mobile apps quickly with Flutter. Flutter is designed as a cross-platform SDK that can be used to build frontend and backend applications in a variety of programming languages including Dart, Java, C/C++, and more. Its powerful framework and tools make it easy to quickly build high-quality mobile apps for any purpose.

2.13 Android Studio

Figure 22: Logo of Android Studio application. [38]

For Android Studio, this program is Google's official Integrated Development Environment (IDE) for building and developing Android applications using IntelliJ IDEA. The goal of Android Studio is to develop an IDE engine that can develop apps on Android more efficiently. GUI design allows app preview in different views on any smartphone model and can display something instantly without having to run the app on an emulator on android feature Studio. []

2.14 Java application

Figure 23: Logo of Java. [39]

Java is an object-oriented programming language that may be used to develop applications for a variety of platforms. It was developed by James A. Gosling and his colleagues at Sun Microsystems, which is now owned by Oracle. The language was intended as an alternative to C++ and is comparable to Objective-C. Java is intended to be highly interoperable and simple to use, allowing programs to be built once and executed on any platform that supports Java without requiring recompilation. This is feasible because the generated code is stored in bytecode, which may operate on any Java virtual machine (JVM) independent of the underlying operating system. Since 2016, Java has been extensively used for client-server web applications, and the produced bytecode may operate on a range of operating systems, including Windows, Linux, and Mac OS.

2.15 Kotlin application

JetBrains created Kotlin, a programming language that has grown in popularity due to its potential to be utilized for Android app development. It is a statically typed language that may be constructed on top of the Java Virtual Machine or JavaScript. It was recently developed natively using LLVM. Kotlin is entirely compatible with Java code, allowing for progressive project transfer.

JetBrains, the maker of prominent IDEs such as IntelliJ and WebStorm, initially launched Kotlin in 2010. It went open-source in 2012, making it a new programming language. Although it was not widely used in its first five years, it has since become a crucial component of mobile application development. When Google announced Kotlin support in 2017, its popularity skyrocketed. Since then, it has been extensively accepted and recommended by 72% of Android application developers.

JetBrains created Kotlin, an open-source programming language that is entirely compatible with Java code. It offers a short learning curve, decreases programming time, supports object-oriented and functional programming, and makes asynchronous programming using coroutines easier. It is cross-platform compatible and very adaptable, allowing developers to work in their own way. In compared to Java, Kotlin's syntax has been streamlined and updated, it addresses Java issues, and it is more secure. Kotlin has grown in popularity among app developers, particularly for Android mobile app development. [40]

2.16 Firebase

Figure 24: Logo of Google Firebase. [41]

Google's Firebase is a development tool that facilitates the creation of iOS, Android, and web apps. It offers a number of services, such as analytics to monitor user activity and enhance app performance, authentication to ensure secure access, cloud messaging to deliver messages across platforms, a real-time database to store and sync data in real-time, Crashlytics to track and resolve stability issues, Performance Monitoring to enhance app performance, and a test lab to test apps on various hardware and configurations. With Firebase, developers can focus on creating better features for their apps rather than spending time debugging issues.

Firebase offers a number of services, including:

1. Analytics - Google Analytics for Firebase provides free, limitless reporting on up to 500 different events. Analytics provides user activity data for iOS and Android apps, enabling improved performance and app marketing decisions.
2. Authentication - The sign-in and onboarding experiences for users are improved by Firebase Authentication, which makes it simple for developers to create secure authentication systems. With support for email and password accounts, phone auth, Google, Facebook, GitHub, Twitter login, and more, this feature provides a comprehensive identity solution.
3. Cloud messaging - Firebase Cloud Messaging (FCM) is a cross-platform messaging tool that lets companies reliably receive and deliver messages on iOS, Android, and the web at no cost.
4. Realtime database - Data may be saved and synchronized in real time across users thanks to the cloud-hosted NoSQL database called Firebase Realtime Database. When

an app is offline, the data is still accessible since it is continuously synchronized across all clients.

5. Crashlytics - A real-time crash reporter called Firebase Crashlytics assists developers in tracking, prioritizing, and resolving stability problems that lower the caliber of their apps. With crashlytics, developers can focus more on creating features for their apps rather than managing and resolving crashes.
6. Performance - Developers may use the Firebase Performance Monitoring tool to get insight into the performance characteristics of their iOS and Android applications and use that information to decide where and when their apps' performance might be enhanced.
7. Test - lab - An infrastructure for testing apps on the cloud is called Firebase Test Lab. Developers may test their iOS or Android apps on a multitude of hardware combinations with a single process. In the Firebase console, they may view the outcomes, including videos, images, and logs.

CHAPTER 3

METHODOLOGY

3.1 Introduction

This chapter explores the notion of deep learning along with its technological infrastructure for data acquisition, and the available literature on machine learning methods. The central theme of the chapter is the COVID-19 pandemic, which is initially examined in a concise manner. The discussion then centers on the critical need to detect infectious diseases at an early stage to minimize their transmission. The present discourse provides a comprehensive overview of the utilization of deep learning techniques in the domain of COVID-19 to facilitate the diagnostic process of crucial physiological parameters.

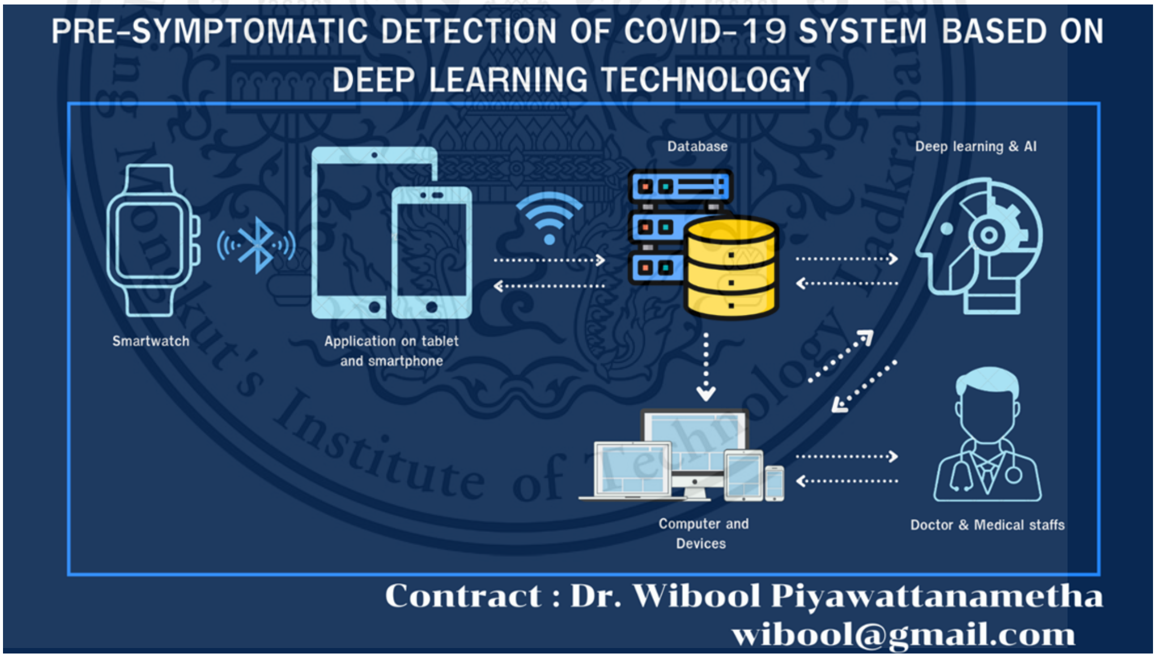


Figure 25: Overview of Pre-symptomatic detection of Covid-19 system based on deep learning technology from smartwatch data.

The process of data acquisition, depicted in the preceding figure, entails the retrieval of essential physiological parameters, such as heart rate, blood oxygen saturation, step count, and temperature, from a smartwatch device through an associated software program. The collated data is subsequently entered into a database, which makes use of an internet connection to forecast a diagnosis of COVID-19. The implementation of this inventive methodology for obtaining data presents a valuable asset in the management of crisis situations.

In addition, the telehealth system has to contact with doctor and give vital data that we got from the patient's watch and from the patient's input. Doctor can analyze from vital signs and diagnosis to find the solutions.

3.2.1 Recruited participants and significant vital sign

The study was carried out in KMITL and conducted with a total of 5300 participants recruited, including 5262 from Stanford paper [8] and 38 students of biomedical engineering KMITL. Details about the parameter can be found in Table. These vital signs in humans are very important for the generalization of the machine-learning model since the gathered data will differ. For example, the mean heart rate value during blood oxygens and sleep cycle

User ID	AverageHeartrate_Jun	AverageHeartrate_Jul	AverageSpO2_Jun	AverageSpO2_Jul
#00001	101	96	98	97
#00002	96	97	98	94
#00003	104	102	99	97
#00004	101	97	97	98
#00005	101	105	95	98
#00006	97	109	94	94

e

Table 1: User ID and their average vital signs per month.

3.2.2 Data acquisition procedure

From the figure overview of pre-symptomatic detection of the Covid-19 system based on deep learning technology from smartwatch data, there are a lot of crucial components to the data collection process. The initial part is smartwatch comes with measurer sensors for vital signs and significant values such as temperature, steps, and time. The secondary is the application due to telehealth needing vital signs display system and the goal of this research is to develop a telehealth system simpler and easier to use. The third is a database system to acquire patient data and linked with AI or artificial intelligence to alert pre-symptomatic Covid-19 through analyzing filled patient symptoms and collected vital signs from the smartwatch

3.2.3 Smartwatch and Application

Each participant wears a Fitbit or J-style, a smartwatch device for the measurements of vital sign parameters, providing integrated blood oxygens, heart rate, step, spO2, and sleep with an optical sensor as we identified in Chapter 2 or depicted in Figure. The data sheet of the vital sign can be found in Appendix. These two smartwatches have different methods to develop applications related to the smartwatch's ability. J-style has a JClife application and a Fitbit mobile application to connect between watch, your mobile phone and its database.

(a) (b)

Figure 26: JClife application (a), and Fitbit mobile application (b).

3.2.3.1 J-Style 2025E Model

The fact that this smartwatch is not using a TCP/IP-based protocol for communication makes it hard to reverse engineer. if it was supposed to be on a local network, things would be a bit easier than just using MAC or IP to sniff and dump packets in a PCAP file to be later analyzed with Wireshark, it could have been cryptic but easy to sniff, Basically a Man in the Middle sort of thing, even a simple CLI tcpdump would also work, but rather it is using Bluetooth which is meant for peer-to-peer networking, means at one time the device can only talk to one master.

An amazing application by Nordic Semiconductor which runs on Android and ios is called NRF Connect which can be used to explore GATT services and characteristics exposed by the device. I can use this app to connect to a smartwatch, know its Unique Address (sort of MAC address), and find which GATT services are exposed by the smartwatch and their corresponding characteristics. First, turn the Bluetooth and smartwatch on and scan for devices. The device was shown up swiftly with the name 'Generic Access' as shown in the screenshot below.

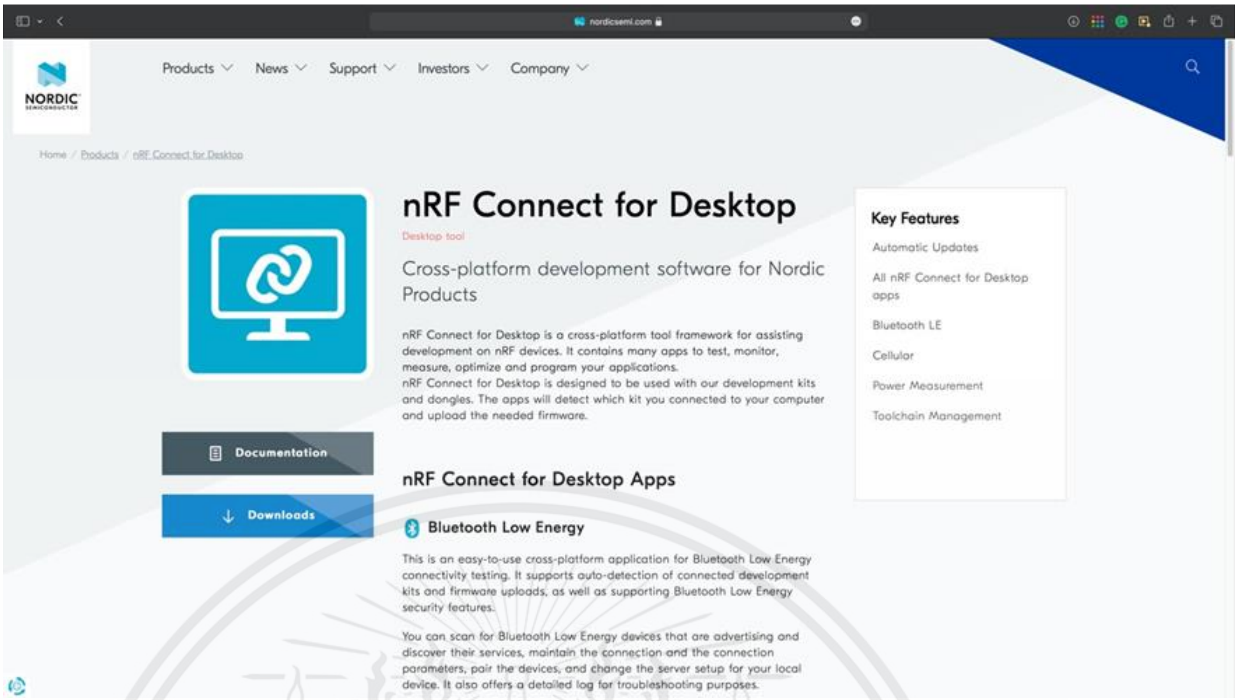


Figure 27: nRF Connect.

Figure 28: Searching Bluetooth devices to connect in nRF Connect.

Connecting to the bulb with the nRF connect app reveals three available GATT services as in the screenshot. two out of three GATT Services exposed by the light bulb are generic to most of BLE Devices, the 0x1800 for Generic access to devices and 0x1801 for Generic pieces of stuff, you can check out the Bluetooth GATT website to find more about this and other GATT services. These services define device names, device types, and statuses as shown in screenshots.

3.2.3.2 Fitbit Model

Fitbit smartwatch is an American electronic company that has products about technology and fitness. The Fitbit smartwatch makes your training regimen as hassle-free as possible by wrapping this little device around your wrist. They are jam-packed with useful functions that make your life easier, including pure pulse, smart track, sleep tracking, and others.

Let's start off by going to the website https://dev.fitbit.com/. This is the website for Fitbit developers, go to manage and select register an app.

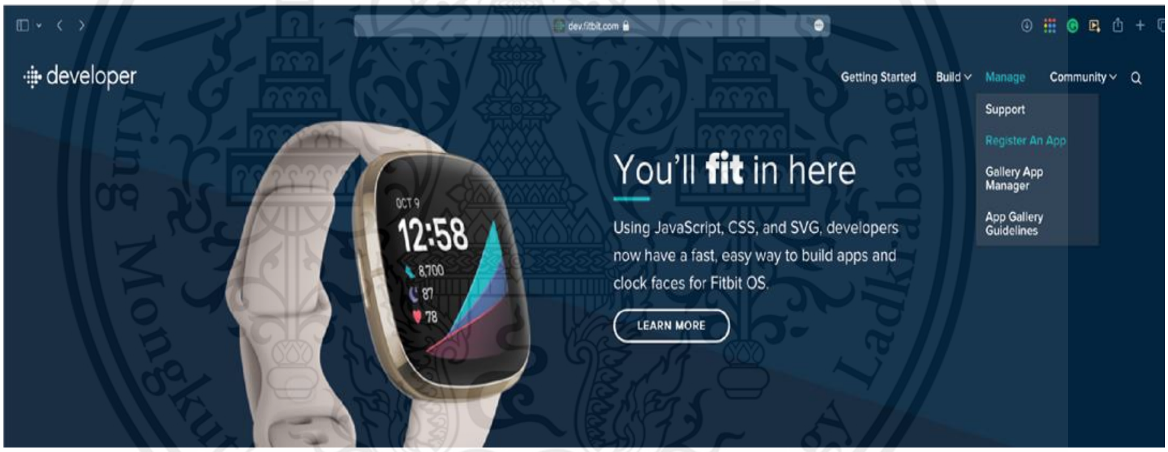


Figure 29: Fitbit developer website and function of registering an app.

After selecting register an app, fill in the information that requires, including application name, description, application website URL, organization, organization website URL, term of service URL, privacy policy URL, OAuth 2.0 application type; for this part, always choose personal, redirect URL, and default access type. After the application has been created, Fitbit will send the application's client ID.

(a) (b)

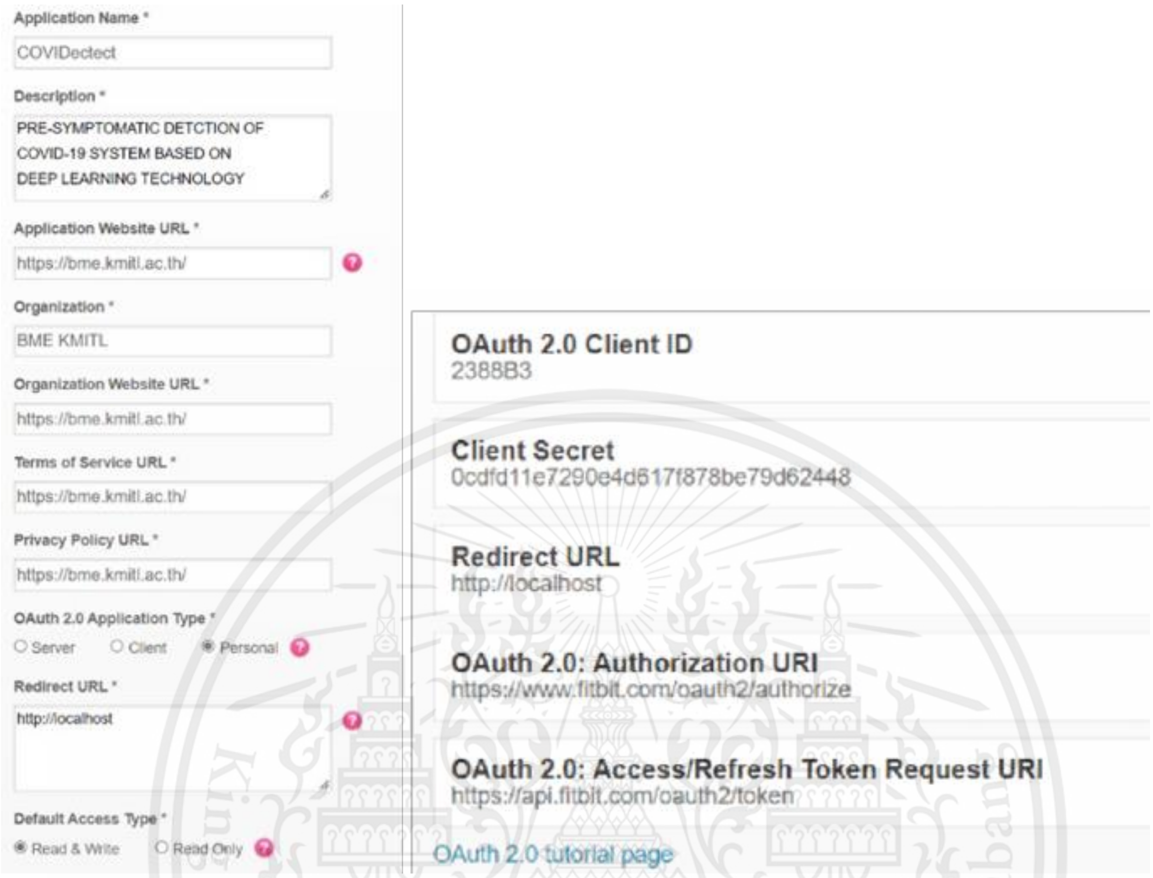


Figure 30: Creating the application on the Fitbit website (a) and creating the application and client ID (b).

The next step is to navigate to the Fitbit developer website at this address: https://dev.fitbit.com/build/reference/web-api/authorization/authorize/. For use with the application, this website has an OAuth 2.0 authorization URL. Choose the implicit grant flow URL. After that, copy the URL and paste it into Notepad++ for separating focused information like client ID and expiration time.

```
Authorization Code Grant Flow with PKCE (recommended)
https://www.fitbit.com/oauth2/authorize?response_type=code
    &client_id=<client ID>
    &redirect_uri=https%3A%2F%2Fexample.com%2Fcallback
    &code_challenge=E9Melhoa2OwvFrEMTJgCHaoeK1t8URWbuGJSstw-cM
    &code_challenge_method=S256
    &scope=activity%20nutrition%20heartrate%20location%20...

Authorization Code Grant Flow
https://www.fitbit.com/oauth2/authorize?response_type=code
    &client_id=<client ID>
    &redirect_uri=https%3A%2F%2Fexample.com%2Fcallback
    &scope=activity%20nutrition%20heartrate%20location%20...

Implicit Grant Flow
https://www.fitbit.com/oauth2/authorize?response_type=token
    &client_id=<client ID>
    &redirect_uri=https%3A%2F%2Fexample.com%2Fcallback
    &expires_in=604800
    &scope=activity%20nutrition%20heartrate%20location%20...
```

Figure 31: The implicit grant flow URL.

The next step, change the client ID in the URL using the client ID that we receive from our application. Then change the redirect URL using the redirect URL of our application. Defined scope of activities of application and duration of application. The reference expiration is shown in this website https://dev.fitbit.com/build/reference/web-api/authorization/authorize/.

The new implicit grant flow URL should now be copied and pasted onto the organization's website as the next step. Afterward, let the app be accessible and get data from the smartwatch. after approval of the application. It would navigate to another website where the application's access token and user id are stored. Save the user id and access token for usage in the code connecting our application to the Fitbit database.

From the Fitbit database, we would use Python language to bring out data by using a user ID and access token to define the watch that we wanted to bring out data. The code structure started with a header to define authorization and token. Next, the request part is the part that defines which type of data we want to bring out for this code.
The last part is for the loop. In this loop, we defined the data type, date type, and printout.

```python
import requests
from openpyxl import Workbook

access_token = 'eyJhbGciOiJIUzI1NiJ9.eyJhdWQiOiIyMzg4QjMiLCJzdWIiOiI5VlRYSjYiLCJpc3MiOiJGaXRiaXQiLCJ0eXAiOiJhY2Nlc3NfdG9rZW4
user_id = '9VTXJ6'
header = {'Authorization': 'Bearer {}'.format(access_token)}


heart_rate_request = requests.get('https://api.fitbit.com/1/user/-/activities/heart/date/2022-06-27/1d.json',headers=header)


print(heart_rate_request.status_code)
data_in=heart_rate_request.json()
```

Figure 32: Using user ID, access token and request for heart rate data to bring out data.

```
for item in data_in['activities-heart-intraday']['dataset']:

    A = f"{data_in['activities-heart'][0]['dateTime']} {item['time']}"
    B = item['value']

    sheet.cell(row=row, column=1).value = A
    sheet.cell(row=row, column=2).value = B
    row += 1

    print(A,B)
```

Figure 33: For loop define the data and date types to print out data.

```
import requests
import time
from pprint import pprint
import json

access_token = 'eyJhbGciOiJIUzI1NiJ9.eyJhdWQiOiIyMzg4QjMiLCJzdWIiOiI5VlRYSjYiLCJpc3MiOiJGaXRiaXQiLCJ0eXAiOiJhY2Nlc3NfdG9rZW
user_id = '9VTXJ6'
header = {'Authorization': 'Bearer {}'.format(access_token)}


activity_request = requests.get('https://api.fitbit.com/1/user/-/activities/steps/date/2022-06-22/1d.json',headers=header)


print(activity_request.status_code)
data_in=activity_request.json()

for data in data_in['activities-steps']:
    print(data_in)
```

Figure 34: Code for bringing steps of activities data per day.

```
import requests
import time
from pprint import pprint
import json

access_token = 'eyJhbGciOiJIUzI1NiJ9.eyJhdWQiOiIyMzg4QjMiLCJzdWIiOiI5VlRYSjYiLCJpc3MiOiJGaXRiaXQiLCJ0eXAiOiJhY2Nlc3NfdG
user_id = '9VTXJ6'
header = {'Authorization': 'Bearer {}'.format(access_token)}


sleep_request = requests.get(' https://api.fitbit.com/1.2/user/-/sleep/date/2022-06-18.json',headers=header)


print(sleep_request.status_code)
data_in=sleep_request.json()

for data in data_in['sleep']:
    print(data)
```

Figure 35: Code for bringing sleep data per day.

```python
import requests
import json
import pandas as pd
from time import sleep
from datetime import datetime

#API connection requires a token supplied by fitbit after an API registration: https://dev.fitbit.com/apps/new
token = 'eyJhbGciOiJIUzI1NiJ9.eyJhdWQiOiIyMzg4QjMiLCJzdWIiOiI5VlRYSjYiLCJpc3MiOiJGaXRiaXQiLCJ0eXAiOiJhY2Nlc3NfdG9r

url_spo2 = 'https://api.fitbit.com/1/user/-/spo2/date/2022-06-07/2022-07-09.json'
response = requests.get(url=url_spo2, headers={'Authorization':'Bearer ' + token}).json()
response
```

Figure 36: Code for bringing spO2 data per day.

3.2.3.3 Java Application

The development process for the Jstyle 2025E app primarily utilizes Android Studio and involves writing code in the Java programming language. The app is designed for use on Android devices, and creating the app involves configuring Gradle and the Android Manifest, establishing communication with Bluetooth Low Energy on smartwatches, and creating layouts and activities within the application.

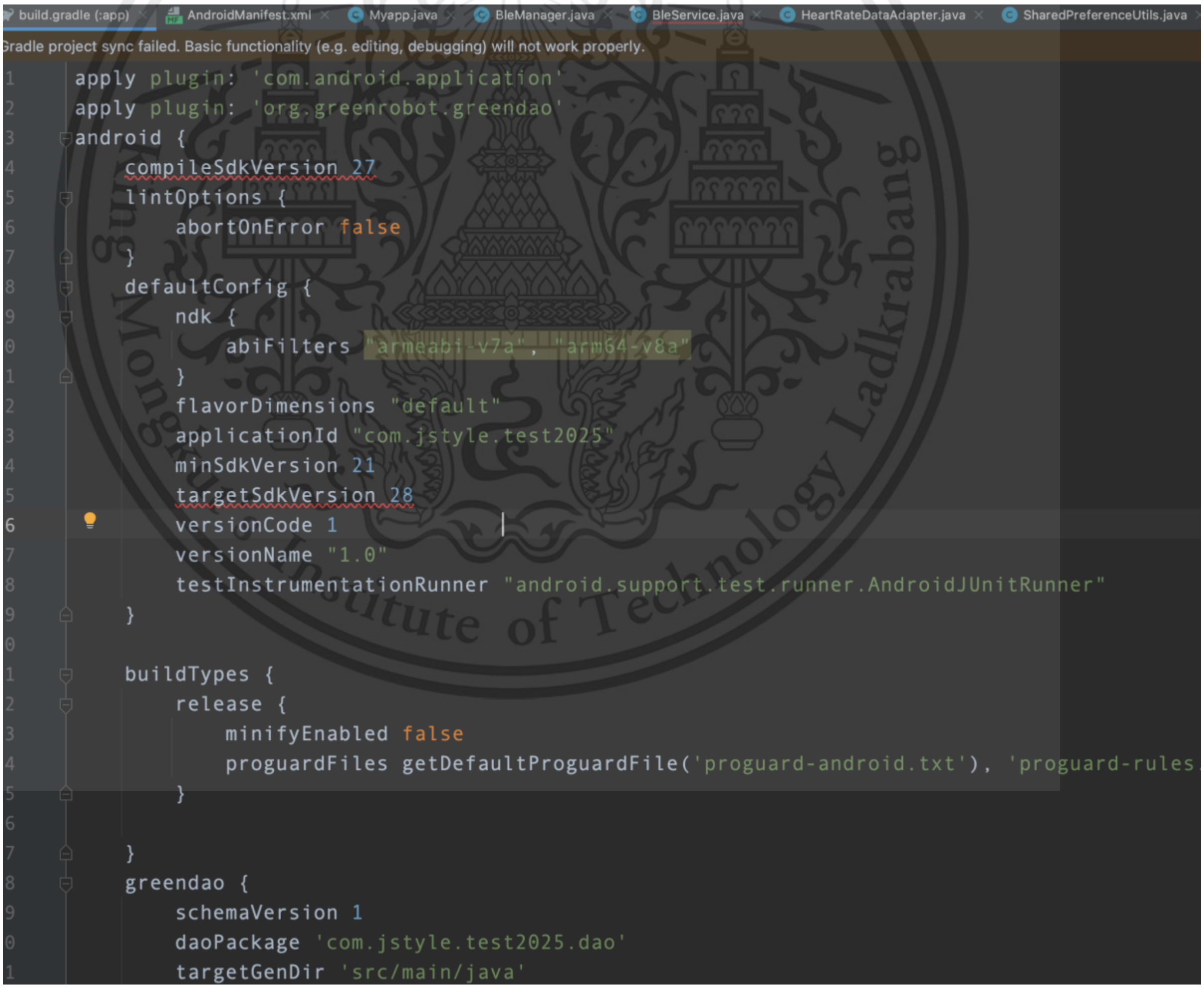
```xml
build.gradle (:app)   AndroidManifest.xml   Myapp.java   BleManager.java   BleService.java   HeartRateDataAdapter.java   SharedPreferenceUtils.java
Gradle project sync failed. Basic functionality (e.g. editing, debugging) will not work properly.
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    package="com.jstyle.test2025">

    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
    <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
    <uses-permission android:name="android.permission.BLUETOOTH" />
    <uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
    <uses-permission android:name="android.permission.SYSTEM_OVERLAY_WINDOW" />
    <uses-permission android:name="android.permission.SYSTEM_ALERT_WINDOW" />
    <uses-permission android:name="android.permission.VIBRATE" />
    <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
    <uses-permission android:name="android.permission.CAMERA" />
    <uses-permission android:name="android.permission.ACCESS_LOCATION_EXTRA_COMMANDS" />
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
    <uses-permission android:name="android.permission.FOREGROUND_SERVICE" />
    <uses-permission android:name="android.permission.BODY_SENSORS" />

    <application
        android:name="com.jstyle.test2025.Myapp"
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="2025test"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme"
        tools:ignore="AllowBackup,GoogleAppIndexingWarning">
```

Figure 37: The example Java code of the Android Manifest file.

The next step in developing the Jstyle 2025E app is setting up the Gradle file. This is a crucial step in which we specify the SDK version and install necessary tools and libraries for app development. In our app, we used API 28 (Android 9.0) to cater to the latest smartphones and set the minimum SDK version to API 21 (Android 5.0) to make the app compatible with older devices. Additionally, we installed required libraries, which are depicted in figure , to aid in the app's development. The Android SDK is a powerful tool that has contributed to the rapid growth of Android apps and contains various programs and libraries required to develop Android apps.

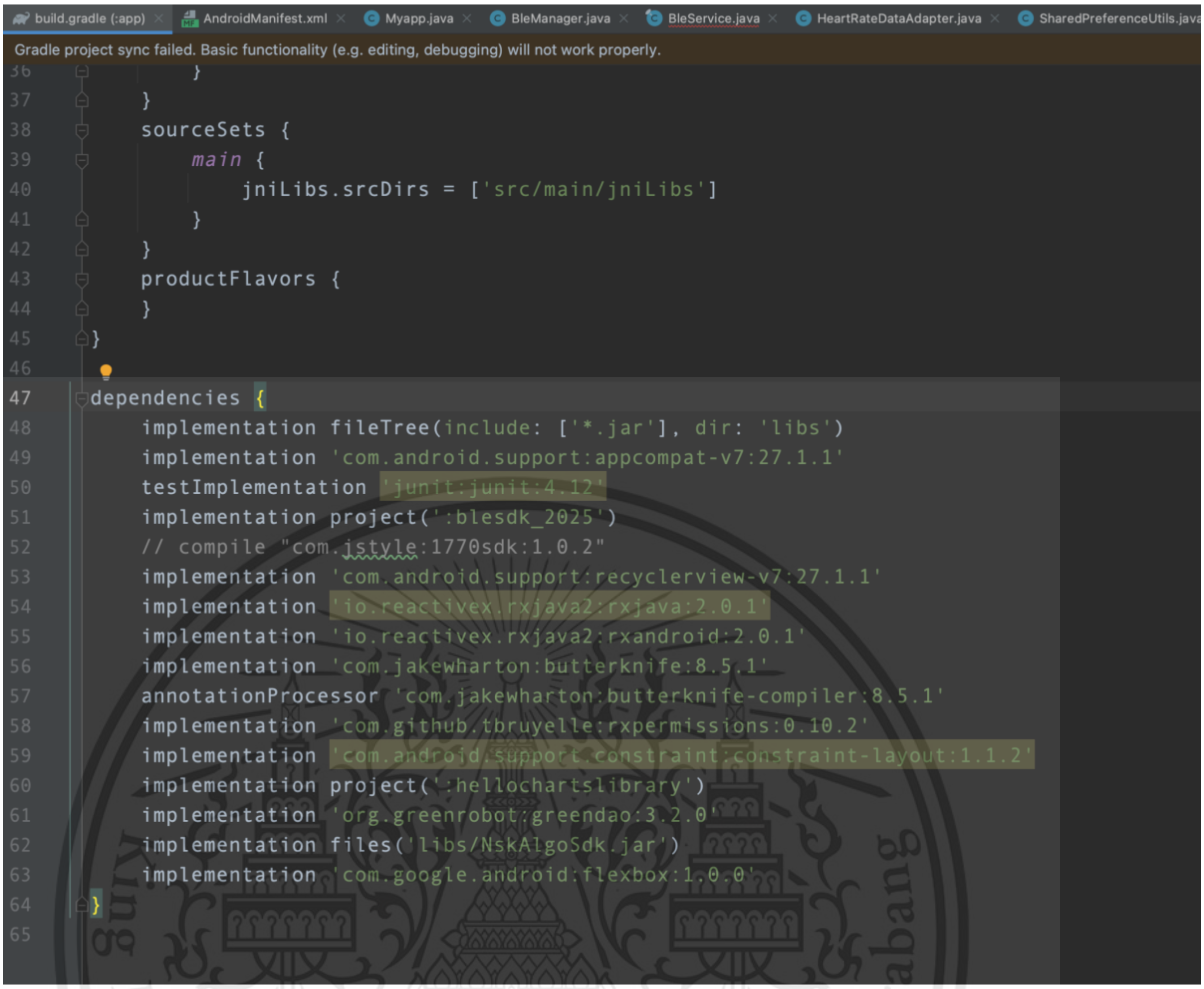
3.2.3.3.1 Setting The Gradle and Android Manifest

The initial step in developing the application involves configuring the Gradle and Android Manifest. This is done by writing Java code (as seen in Figure) to include crucial permissions like Internet access, defining the application name and logo, and specifying which activities will be part of the application.



```
build.gradle (:app)   AndroidManifest.xml   Myapp.java   BleManager.java   BleService.java   HeartRateDataAdapter.java   SharedPreferenceUtils.java
Gradle project sync failed. Basic functionality (e.g. editing, debugging) will not work properly.
apply plugin: 'com.android.application'
apply plugin: 'org.greenrobot.greendao'
android {
    compileSdkVersion 27
    lintOptions {
        abortOnError false
    }
    defaultConfig {
        ndk {
            abiFilters "armeabi-v7a", "arm64-v8a"
        }
        flavorDimensions "default"
        applicationId "com.jstyle.test2025"
        minSdkVersion 21
        targetSdkVersion 28
        versionCode 1
        versionName "1.0"
        testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"
    }

    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules
        }

    }
    greendao {
        schemaVersion 1
        daoPackage 'com.jstyle.test2025.dao'
        targetGenDir 'src/main/java'
```

Figure 38: The version of SDK in Jstyle 2025E app.



```
build.gradle (:app)   AndroidManifest.xml   Myapp.java   BleManager.java   BleService.java   HeartRateDataAdapter.java   SharedPreferenceUtils.java
Gradle project sync failed. Basic functionality (e.g. editing, debugging) will not work properly.
            }
        }
        sourceSets {
            main {
                jniLibs.srcDirs = ['src/main/jniLibs']
            }
        }
        productFlavors {
        }
}

dependencies {
    implementation fileTree(include: ['*.jar'], dir: 'libs')
    implementation 'com.android.support:appcompat-v7:27.1.1'
    testImplementation 'junit:junit:4.12'
    implementation project(':blesdk_2025')
    // compile "com.jstyle:1770sdk:1.0.2"
    implementation 'com.android.support:recyclerview-v7:27.1.1'
    implementation 'io.reactivex.rxjava2:rxjava:2.0.1'
    implementation 'io.reactivex.rxjava2:rxandroid:2.0.1'
    implementation 'com.jakewharton:butterknife:8.5.1'
    annotationProcessor 'com.jakewharton:butterknife-compiler:8.5.1'
    implementation 'com.github.tbruyelle:rxpermissions:0.10.2'
    implementation 'com.android.support.constraint:constraint-layout:1.1.2'
    implementation project(':hellochartslibrary')
    implementation 'org.greenrobot:greendao:3.2.0'
    implementation files('libs/NskAlgoSdk.jar')
    implementation 'com.google.android:flexbox:1.0.0'
}
```

Figure 39: The list of libraries that were installed.

3.2.3.3.2 Connection Between Jstyle 2025E Application and Bluetooth Low Energy (BLE)

For BLE-enabled devices to connect with the Jstyle 2025E smartwatch and its application, a communication channel must be built after setting the Android Manifest and Gradle. As seen in Figure, the Android Manifest file must declare a number of permissions in order to use the Bluetooth Low Energy APIs. The BluetoothAdapter must be consulted, as shown in Figure, to confirm whether the device supports Bluetooth after receiving authorization to use it. The device searches for nearby Bluetooth Low Energy (BLE) devices if Bluetooth is enabled. As shown in Figure, the device then finds another device with the same UUID, which is a special collection of numbers used to identify the device and prove that the hardware utilized in our project is special. The device then creates a connection with the GATT server of the BLE device to learn more about its capabilities. Based on the connected device's services and characteristics, data can be sent to and received from it after a connection has been made.

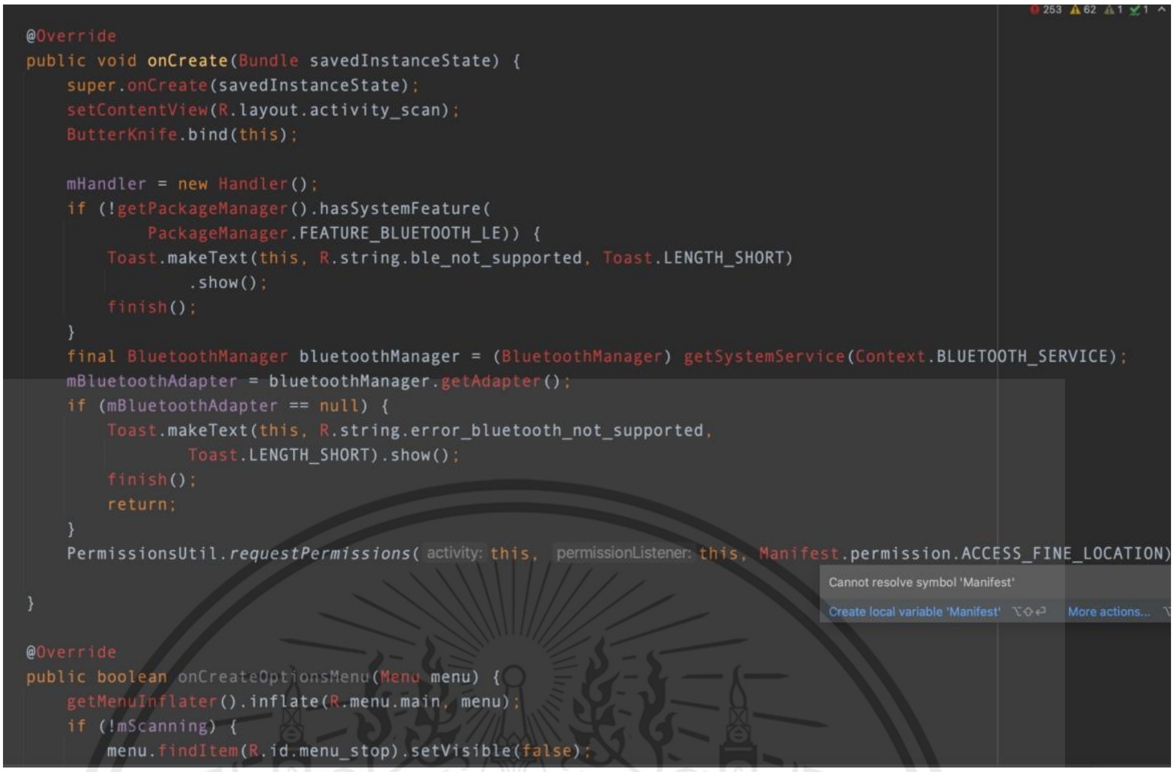
To scan for devices that match the UUID, we will utilize a code command to scan and list the device names, connect to the desired device, and retrieve data. In the event that a device has a UUID that does not match a UUID on the app, the device's name will be displayed as an unknown device, as illustrated in Figure. These meticulous steps ensure that our app communicates effectively with BLE-enabled devices and provides our users with seamless functionality.

```xml
    <uses-permission android:name="android.permission.BLUETOOTH" />
    <uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
```

Figure 40: Bluetooth Permission in Android Manifest.

```java


11 usages
public final class BleService extends Service {
    5 usages
    private static final String TAG = "BleService";
    1 usage
    private static final UUID NOTIY = UUID
            .fromString("00002902-0000-1000-8000-00805f9b34fb");
    2 usages
    private static final UUID SERVICE_DATA = UUID
            .fromString("0000fff0-0000-1000-8000-00805f9b34fb");
    1 usage
    private static final UUID DATA_Characteristic = UUID
            .fromString("0000fff6-0000-1000-8000-00805f9b34fb");
    1 usage
    private static final UUID NOTIY_Characteristic = UUID
            .fromString("0000fff7-0000-1000-8000-00805f9b34fb");
    3 usages
    private boolean NeedReconnect=false;
    2 usages
    public final static String ACTION_GATT_onDescriptorWrite = "com.jstylelife.ble.service.onDescriptorWrite";
    1 usage
    public final static String ACTION_GATT_CONNECTED = "com.jstylelife.ble.service.ACTION_GATT_CONNECTED";
    3 usages
    public final static String ACTION_GATT_DISCONNECTED = "com.jstylelife.ble.service.ACTION_GATT_DISCONNECTED";
    3 usages
    public final static String ACTION_DATA_AVAILABLE = "com.jstylelife.ble.service.ACTION_DATA_AVAILABLE";
    public HashMap<BluetoothDevice, BluetoothGatt> hasp = new HashMap<BluetoothDevice, BluetoothGatt>();
    1 usage
    private final IBinder kBinder = new LocalBinder();
```

Figure 41: UUID of Jstyle model.



```java
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_scan);
    ButterKnife.bind(this);

    mHandler = new Handler();
    if (!getPackageManager().hasSystemFeature(
            PackageManager.FEATURE_BLUETOOTH_LE)) {
        Toast.makeText(this, R.string.ble_not_supported, Toast.LENGTH_SHORT)
                .show();
        finish();
    }
    final BluetoothManager bluetoothManager = (BluetoothManager) getSystemService(Context.BLUETOOTH_SERVICE);
    mBluetoothAdapter = bluetoothManager.getAdapter();
    if (mBluetoothAdapter == null) {
        Toast.makeText(this, R.string.error_bluetooth_not_supported,
                Toast.LENGTH_SHORT).show();
        finish();
        return;
    }
    PermissionsUtil.requestPermissions( activity: this,  permissionListener: this, Manifest.permission.ACCESS_FINE_LOCATION)

}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
    getMenuInflater().inflate(R.menu.main, menu);
    if (!mScanning) {
        menu.findItem(R.id.menu_stop).setVisible(false);
```

Figure 42: Java code for scanning the list of device’s names by connecting BLE.

3.2.3.3.3 Layouts of Application

The layouts of the app are programmed in Java using Android Studio. These layouts include the list of device names that appears when the program scans Bluetooth Low Energy (BLE), the layout for the list of data, and the main page layout that shows the list of functions and real-time monitoring. With the goal of offering an aesthetically beautiful and user-friendly experience, all layouts are produced in.xml files.

3.2.3.3.4 Switching to Kotlin: Overcoming Compatibility Issues and Embracing Improved Safety and Efficiency in Android App Development

Many Java-based app developers discovered that their apps were crashing as a result of compatibility difficulties when the latest Android version was published. Developers found this annoying since it required them to rewrite their programs in Kotlin, a new language. For many years, Java has been the most popular language for creating Android apps, but Kotlin has recently become increasingly common because of its enhanced safety features and simpler syntax. Developers may utilize both Kotlin and Java in the same project because they are both fully compatible. Making the conversion from Java to Kotlin might be difficult for developers at first. Kotlin makes it simpler to develop and maintain code because to its clear syntax and minimal boilerplate. Null pointer exceptions, a typical cause of errors in Java code, may be avoided thanks to Kotlin's null safety capabilities. Developers must take into account these differences while rebuilding an app in Kotlin. Developers must exercise greater caution when handling null values, for instance, due to Kotlin's use of nullable and non-nullable types. In order to lessen code duplication, Kotlin also offers a set of extension methods that may be used to enhance the functionality of already existing classes.Many developers have discovered that migrating to Kotlin is worthwhile despite some early difficulties. It may be possible to write less code and maintain the app more easily in the long term using Kotlin's enhanced safety features and compact syntax.

In conclusion, switching to Kotlin can ultimately be a good choice, even though the compatibility concerns brought on by the recent Android upgrade may be annoying for Java developers. The more streamlined syntax and enhanced safety features of Kotlin make it simpler to write and maintain code, yet the language is completely compatible with Java, enabling developers to utilize both in the same project. Developers may produce more dependable and effective apps that work with next Android releases by using Kotlin.

3.2.3.4 Kotlin Application

Our primary goal for this project is to create two telehealth systems and applications for smartwatch bands. We will work specifically with the Fitbit sense model and the Jstyle model, both of which have distinctive qualities that will influence how we develop. The distinctions between these two models will be taken into account by our staff as they employ techniques that are most appropriate for their skill sets.

The Jstyle 2025E model app is being developed mostly with Android Studio and the Kotlin programming language. Our group concentrates on developing an app that is especially suited to the Android and iOS operating systems. We perform a number of important actions to do this, such as configuring Gradle and the Android Manifest, connecting the wristwatch to Bluetooth Low Energy, and developing the appropriate layouts and activities for the program. By closely adhering to this development process, we make sure that our software is both useful and easy to use.

3.2.3.4.1 Setting The Gradle and Android Manifest

Configuring Gradle and the Android Manifest is the first step in generating and developing the application. Writing Java code is required for this procedure, as shown in Figure, to include crucial permissions, such as Internet permission, define the application's name, establish the application's logo, and guarantee the smooth integration of all application-related operations. These steps are essential to ensure the program runs without a hitch and performs to its full potential. Our team can produce a useful, high-caliber application that satisfies the expectations of our customers by following this meticulous development methodology.

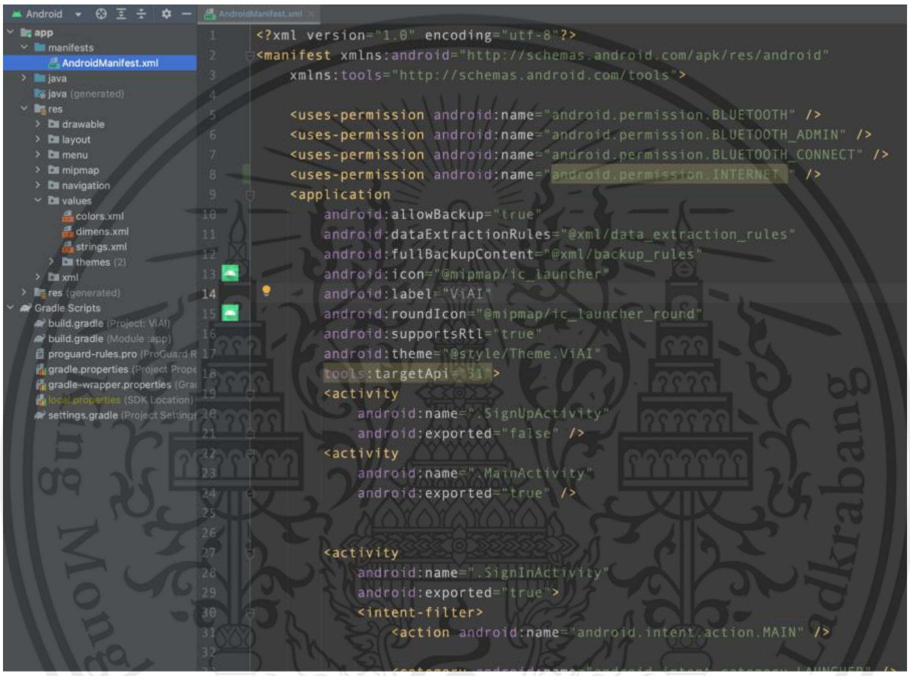


Figure 43: The example Kotlin code of the Android Manifest file.

The configuration of Gradle, as shown in Figure, is an essential step in creating our application. The Software Development Kit (SDK) version of the program may be changed using this file. The Google SDK gives programmers a tool to make programs and applications for the Android operating system that are free for the public to download and use. This has aided in the quick expansion of Android applications. The Jstyle 2025E app was developed for the most recent devices, thus in order to support them, we added API 33 (Android 13). Additionally, we set the SDK's minimal need so that older smartphones running Android 11.0 or API 31 may still use our app. Additionally, we set up libraries, which are crucial resources for creating this application and are shown in Figure. These painstaking procedures guarantee that our software works flawlessly for our consumers across a broad range of devices.

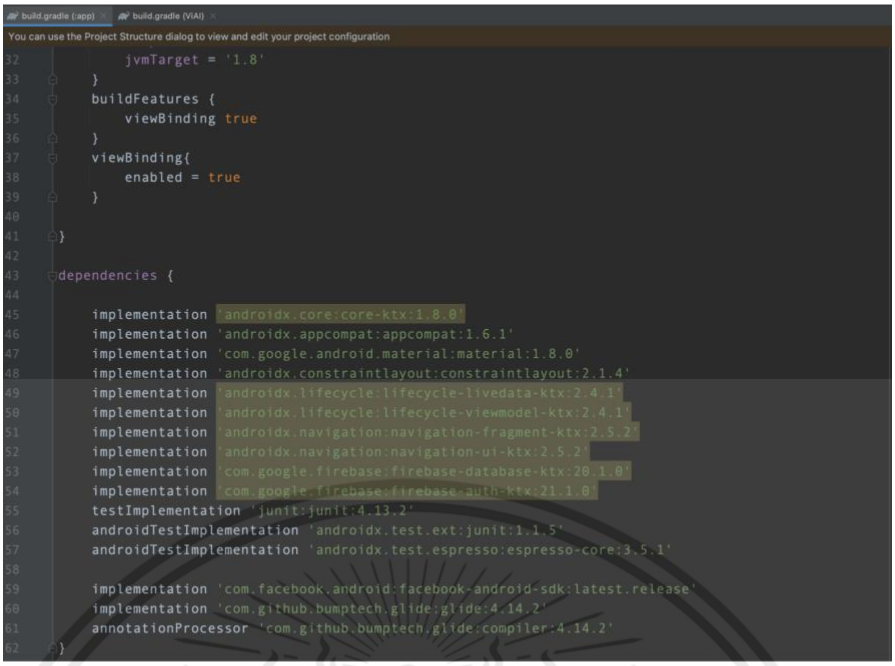


Figure 44: The version of SDK in the J-style 2025E app.

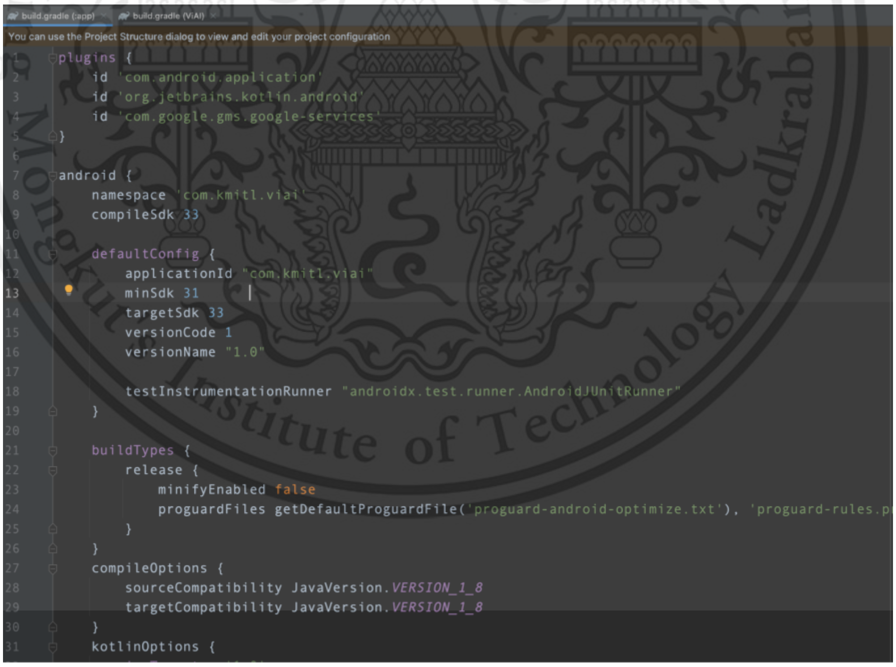


Figure 45: The list of libraries that were installed.

3.2.3.4.2 Connection Between Jstyle 2025E Application and Bluetooth Low Energy (BLE)

For BLE-enabled devices to connect with the Jstyle 2025E smartwatch and its application, a communication channel must be built after setting the Android Manifest and Gradle. As seen in Figure, the Android Manifest file must declare a number of permissions in order to use the Bluetooth Low Energy APIs. The BluetoothAdapter must be consulted, as shown in Figure, to confirm whether the device supports Bluetooth after receiving authorization to use it. The device searches for nearby Bluetooth Low Energy (BLE) devices if Bluetooth is enabled. As shown in Figure, the device then finds another device with the same UUID, which is a special collection of numbers used to identify the device and prove that the hardware utilized in our project is special. The device then creates a connection with the GATT server of the BLE device to learn more about its capabilities. Based on the connected device's services and characteristics, data can be sent to and received from it after a connection has been made.

We will use a code command to scan and list the device names, connect to the target device, and get data from any devices that match the UUID. A device's name will be presented as an unknown device in the case that its UUID does not match one in the app, as seen in Figure. These painstaking measures guarantee that our program works flawlessly for users and properly connects with BLE-enabled devices.

build.gradle (:app) | AndroidManifest.xml | build.gradle (ViAI)

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">

    <uses-permission android:name="android.permission.BLUETOOTH" />
    <uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
    <uses-permission android:name="android.permission.BLUETOOTH_CONNECT" />
```

Figure 46: Bluetooth Permission in Android Manifest.

```
val deviceEditText: EditText = binding.editTextDevice
val connectButton: Button = binding.buttonConnect
val disconnectButton: Button = binding.buttonDisconnect

val listView: ListView = binding.deviceListView
deviceListAdapter = ArrayAdapter(
    requireContext(),
    android.R.layout.simple_list_item_1,
    mutableListOf()
)
listView.adapter = deviceListAdapter
listView.setOnItemClickListener { _, _, position, _ ->
    val deviceName = deviceListAdapter.getItem(position)
    textView.text = "Connecting to $deviceName..."
    val device = if (ActivityCompat.checkSelfPermission(
            requireContext(),
            Manifest.permission.BLUETOOTH_CONNECT
        ) != PackageManager.PERMISSION_GRANTED
    ) {
        ActivityCompat.requestPermissions(
            requireActivity(),
            arrayOf(Manifest.permission.BLUETOOTH_CONNECT),
            REQUEST_PERMISSION_CODE
        )
        return@setOnItemClickListener
    } else {
        val device = bluetoothAdapter.bondedDevices.find { it.name == dev
        if (device != null) {
            bluetoothGatt = device.connectGatt(requireContext(), autoConne
        } else {
            textView.text = "Device $deviceName not found"
        }
    }
```

Figure 47: Java code for requesting enable Bluetooth.

```
companion object {
    private const val REQUEST_PERMISSION_CODE = 1
    private val SERVICE_UUID = UUID.fromString( name: "0000ffe0-0000-1000-8000-00805f9b34fb")
    private val CHARACTERISTIC_UUID = UUID.fromString( name: "0000ffe1-0000-1000-8000-00805f9b34fb")
```

Figure 48: UUID of Jstyle model.

```
private val gattCallback = object : BluetoothGattCallback() {
    //อ่านค่าการเปลี่ยนแปลง
    override fun onConnectionStateChange(gatt: BluetoothGatt, status: Int, newState: Int) {
        if (newState == BluetoothProfile.STATE_CONNECTED) {
            if (ActivityCompat.checkSelfPermission(
                    requireContext(),
                    Manifest.permission.BLUETOOTH_CONNECT
                ) != PackageManager.PERMISSION_GRANTED
            ) {
                // Permission not granted, request it
                ActivityCompat.requestPermissions(
                    requireActivity(),
                    arrayOf(Manifest.permission.BLUETOOTH_CONNECT),
                    REQUEST_PERMISSION_CODE
                )
                return
            }
            // Permission granted, discover services
            gatt.discoverServices()
        }
    }
```

```
private const final val REQUEST_PERMISSION_CODE: Int
com.kmitl.viai.ui.home.HomeFragment.Companion
ViAI.app.main
```

Figure 49: Function gattCallback for monotor bluetoothGatt

```
    override fun onServicesDiscovered(gatt: BluetoothGatt, status: Int) {
        val service = gatt.getService(SERVICE_UUID)
        characteristic = service.getCharacteristic(CHARACTERISTIC_UUID)
        if (ActivityCompat.checkSelfPermission(
                requireContext(),
                Manifest.permission.BLUETOOTH_CONNECT
            ) != PackageManager.PERMISSION_GRANTED
        ) {
            // Permission not granted, request it
            ActivityCompat.requestPermissions(
                requireActivity(),
                arrayOf(Manifest.permission.BLUETOOTH_CONNECT),
                REQUEST_PERMISSION_CODE
            )
        } else {
            // Permission already granted, set characteristic notification
            gatt.setCharacteristicNotification(characteristic, enable: true)
        }
    }
```

Figure 50: onServiceDiscovered function

```
    override fun onCharacteristicChanged(
        gatt: BluetoothGatt,
        characteristic: BluetoothGattCharacteristic
    ) {
        val textView: TextView = binding.textHome
        val value = characteristic.getStringValue( offset: 0)
        requireActivity().runOnUiThread {
            textView.text = value
        }
    }
}
```

Figure 51: onCharacteristicChanged function

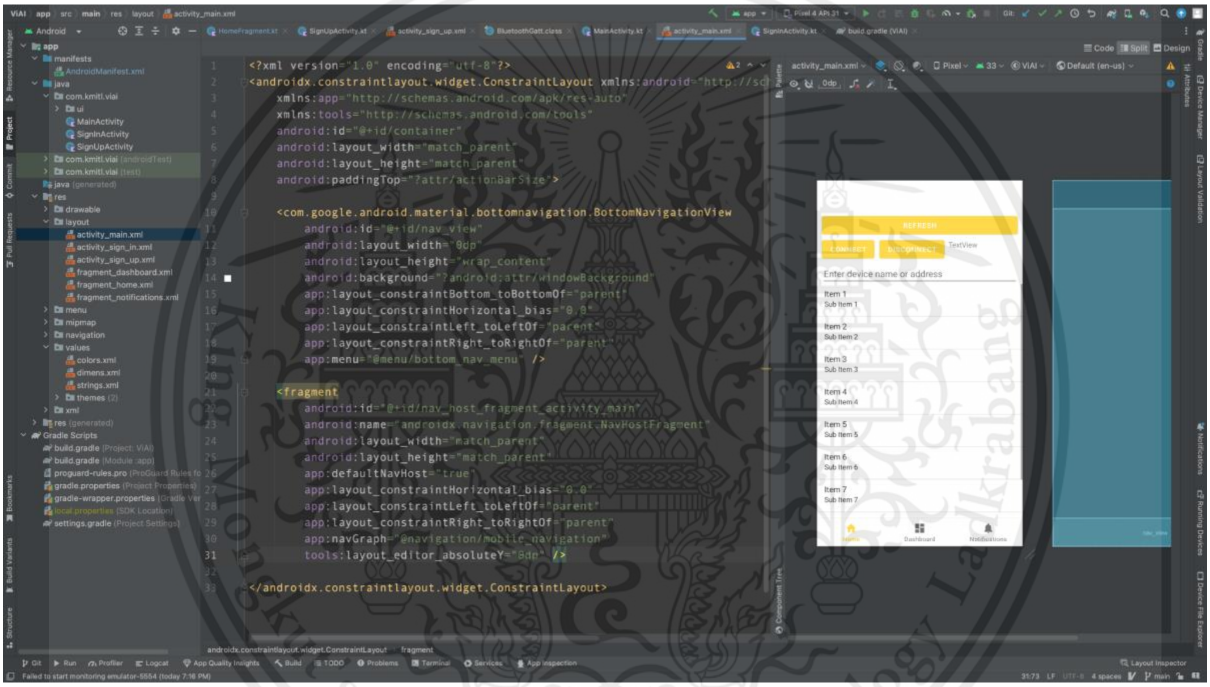
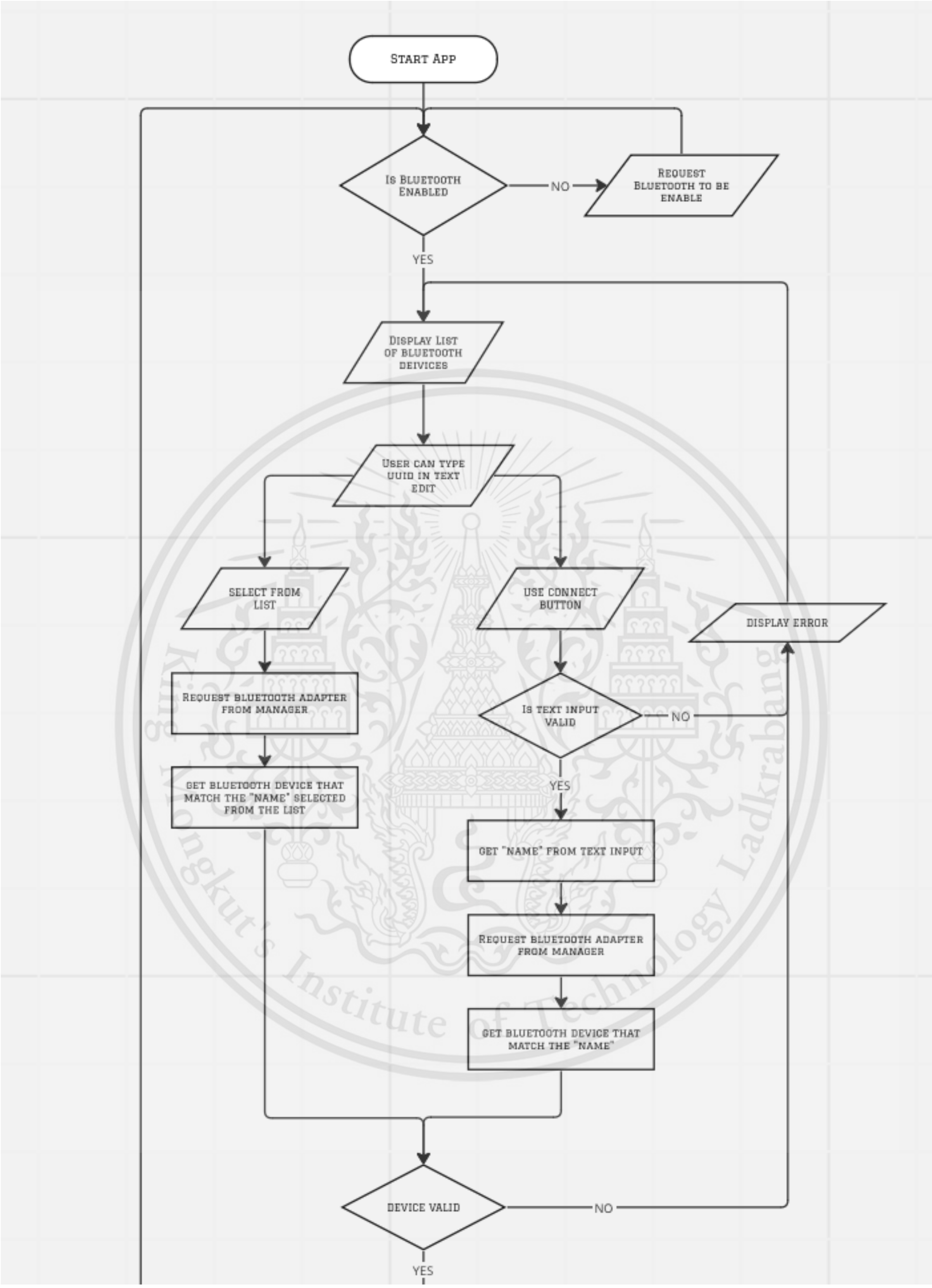


Figure 52: Layout of bluetooth connecting page



START APP
IS BLUETOOTH ENABLED
NO
REQUEST BLUETOOTH TO BE ENABLE
YES
DISPLAY LIST OF BLUETOOTH DEIVICES
USER CAN TYPE UUID IN TEXT EDIT
SELECT FROM LIST
USE CONNECT BUTTON
DISPLAY ERROR
REQUEST BLUETOOTH ADAPTER FROM MANAGER
IS TEXT INPUT VALID
NO
GET BLUETOOTH DEVICE THAT MATCH THE "NAME" SELECTED FROM THE LIST
YES
GET "NAME" FROM TEXT INPUT
REQUEST BLUETOOTH ADAPTER FROM MANAGER
GET BLUETOOTH DEVICE THAT MATCH THE "NAME"
DEVICE VALID
NO
YES

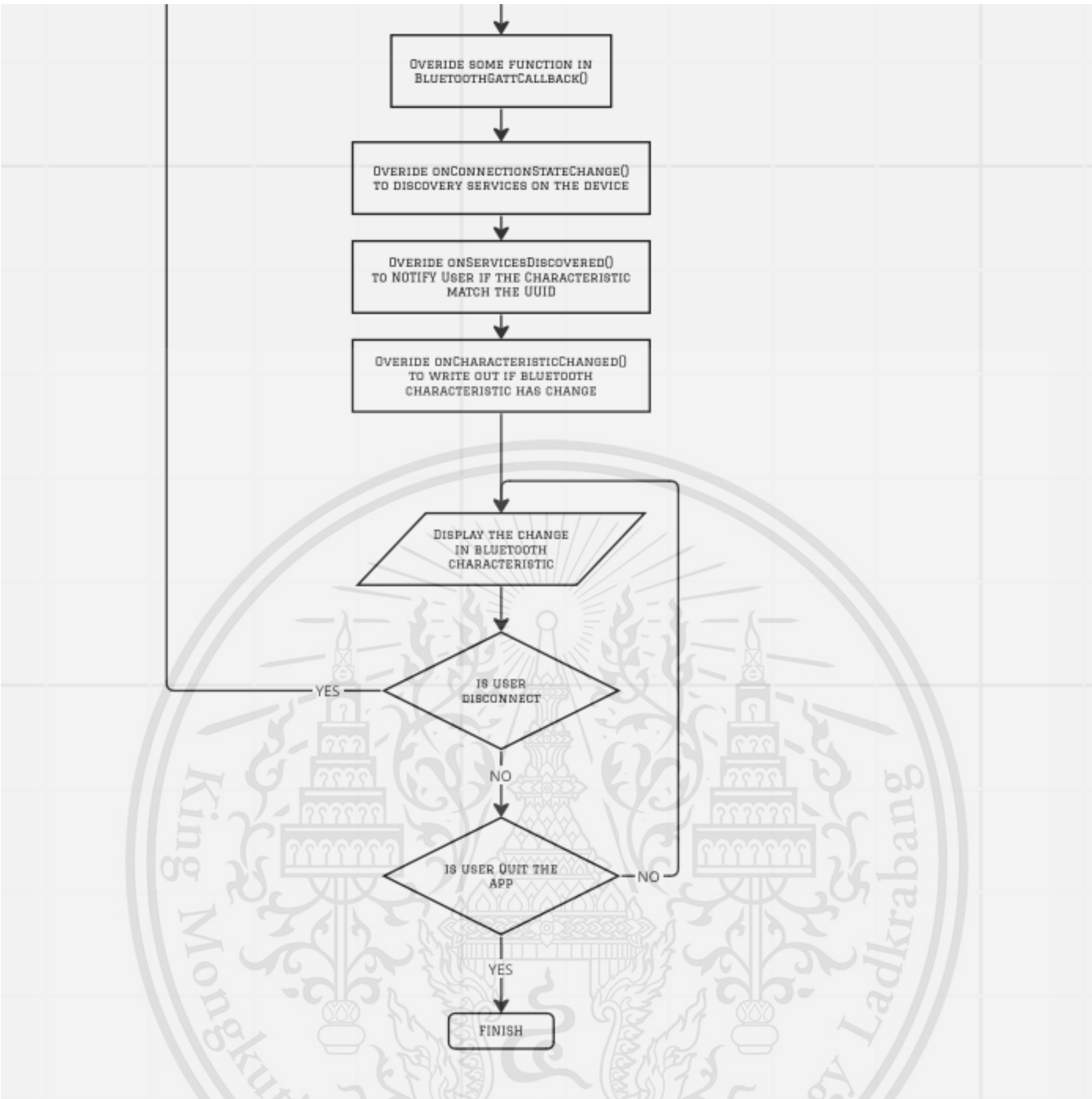


Figure 53: Flow chart of connecting and acquired data form bluetooth

1. The user opens the application on their device.
2. The application checks whether the user is logged in or not. If the user is not logged in, the application prompts the user to log in or register a new account.
3. After the user logs in or registers a new account, the application displays the main menu to the user.
4. The user selects the "Create a new event" option from the main menu.
5. The application prompts the user to enter the details of the event, such as the event name, date, time, location, and description.
6. The user enters the event details and submits the form.
7. The application saves the event details to the database and displays a confirmation message to the user.

8. The user can view the newly created event in the "My Events" section of the application.
9. If the user selects the "View all events" option from the main menu, the application retrieves all events from the database and displays them to the user.
10. The user can select a specific event to view more details about it.
11. If the user selects the "Edit" option for an event, the application allows the user to edit the event details and save the changes to the database.
12. If the user selects the "Delete" option for an event, the application prompts the user to confirm the deletion and then removes the event from the database.

3.2.3.4.3 Login and Registration system

3.2.3.4.3.1 Login and Register Layouts of Application

"SignInActivity.kt" is the name of the class file, which serves as the principal point of entry for the Android application under development. The class file's code comprises a number of methods and functions that define the app's functionality and behavior in relation to the activity_sign_in.xml file. The code editor window is divided into two portions, with the right side displaying the relevant layout XML file and the left side displaying the XML code. The visual elements of the program, such as buttons and text boxes, are defined in the layout file. At the bottom of the IDE, a terminal window provides details about the app sign-in.

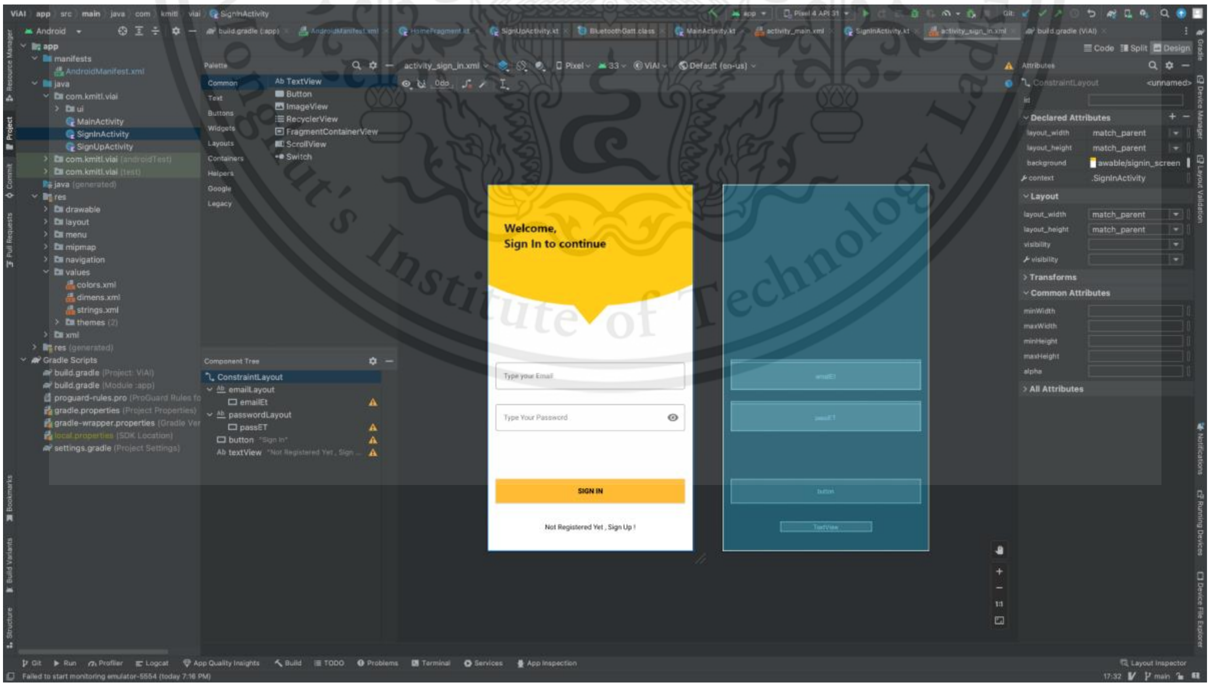
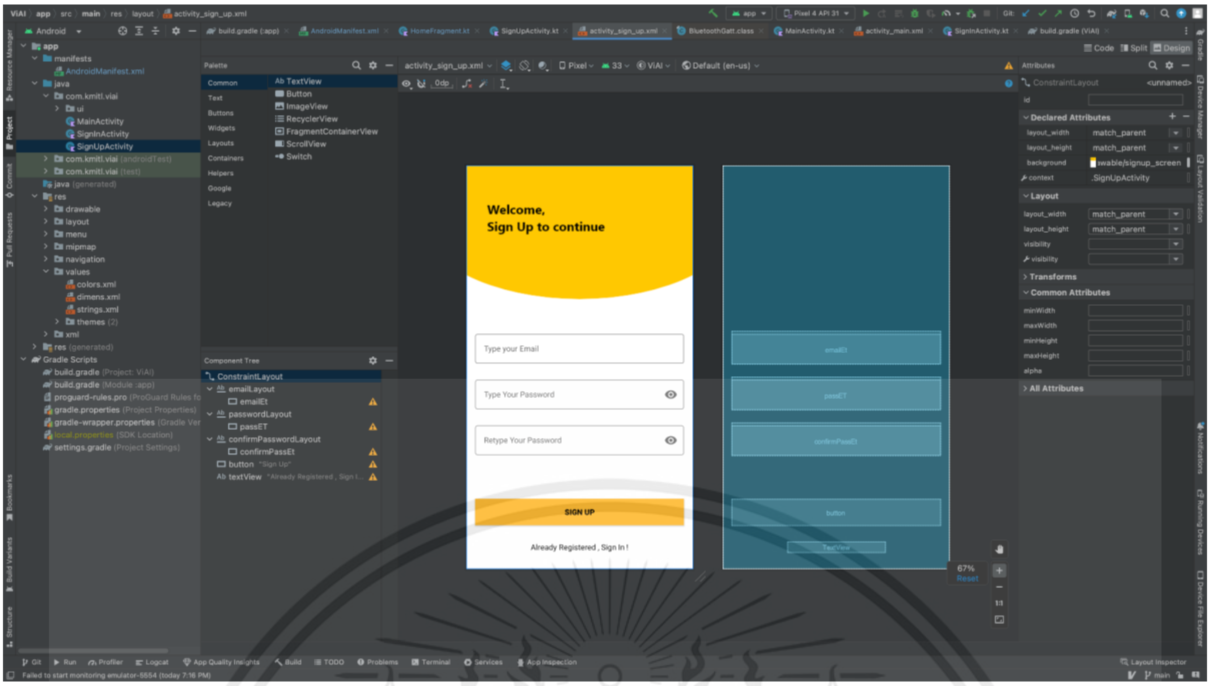


Figure 54: The layout of login



\

Figure 55: The layout of register

3.2.3.4.3.2 Login and Register activities.

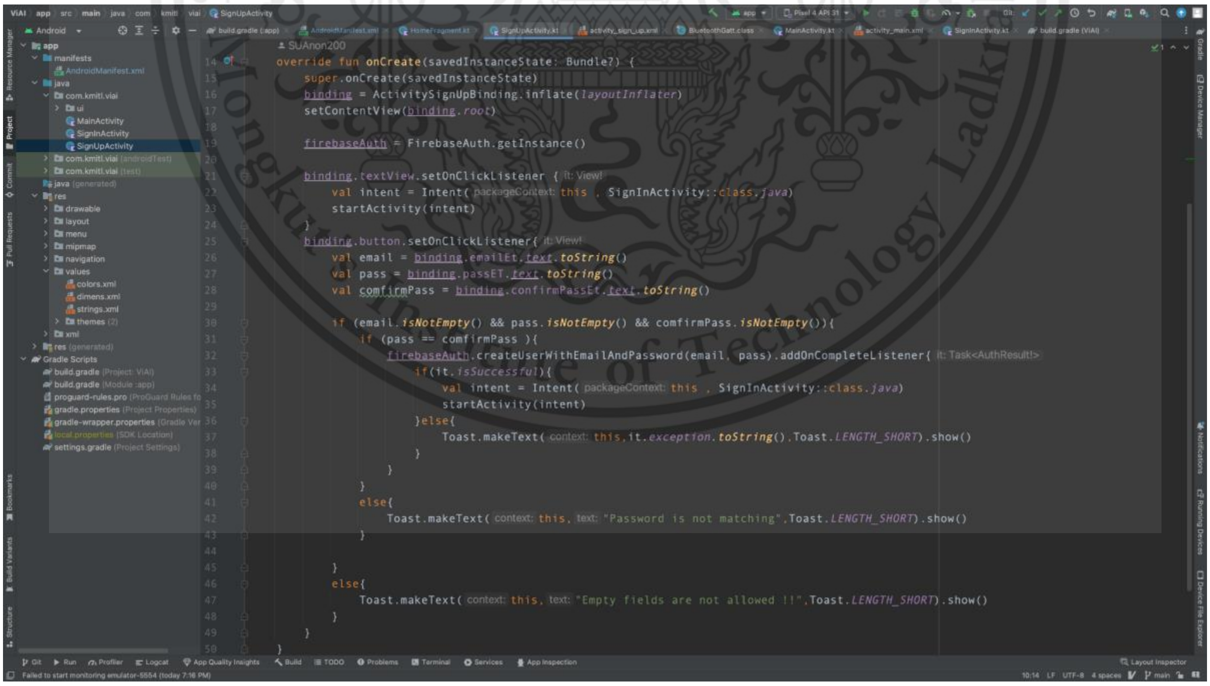


Figure 56: Sign up activities file.

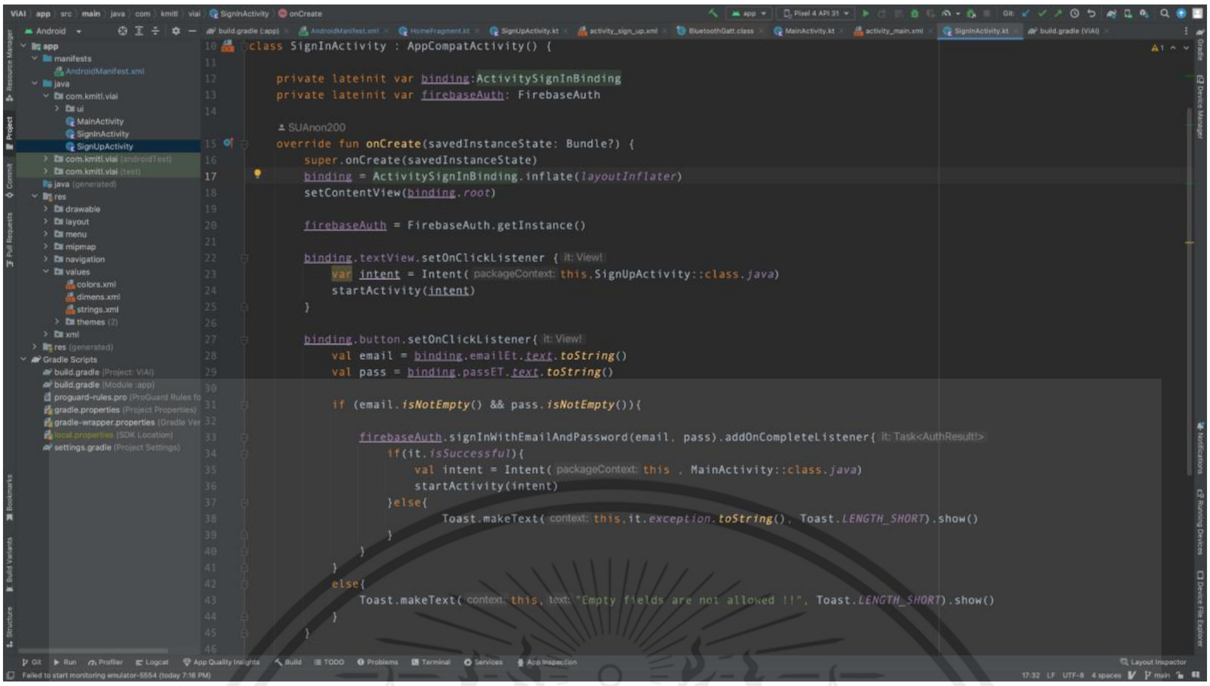


Figure 57: Sign in activities file

3.2.3.4.4 Firebase and Realtime database

Connecting an app to Firebase involves creating a project in the Firebase console, configuring the app in the project, and adding the necessary Firebase dependencies to the app. Once the app is connected to Firebase, it can use Firebase services such as the Realtime Database, Cloud Firestore, and Firebase Authentication.

Adding the Firebase Authentication SDK to an application involves including the Firebase Authentication dependencies in the app's build.gradle file and initialize the Firebase Authentication object in the app's code. Firebase Authentication provides a secure way for users to sign up and sign in to an app using email and password authentication, phone number authentication, and third-party authentication providers such as Google and Facebook. With the Firebase Authentication SDK, developers can easily add authentication functionality to their apps and manage user authentication states.

```
import com.google.firebase.auth.FirebaseAuth
```

Figure 58: Import firebase to activity

```
private lateinit var binding: ActivitySignUpBinding
private lateinit var firebaseAuth: FirebaseAuth
SUAnon200
```

Figure 59: Declare a variable for Firebase authentication

```
firebaseAuth.createUserWithEmailAndPassword(email, pass).addOnCompleteListener{ it: Task<AuthResult!>
    if(it.isSuccessful){
        val intent = Intent( packageContext: this , SignInActivity::class.java)
        startActivity(intent)
    }else{
        Toast.makeText( context: this,it.exception.toString(),Toast.LENGTH_SHORT).show()
    }
```

Figure 60: Sending user authentication to Firebase

3.2.4 Abnormal detection

3.2.4.1 Abnormal detection using seasonal decomposition

The utilization of seasonal decomposition for the purpose of anomaly detection is a statistical procedure that seeks to identify any deviations or anomalies that may be present in the temporal series data. The proposed approach entails the decomposition of a temporal dataset into constituent seasonal, trending, and residual parts through the utilization of mathematical modeling techniques, such as the Seasonal Time Series Separation (STL) algorithm. It is important to capitalize the first letter of each word in a sentence. This helps to provide clarity and readability to the text. Additionally, using proper grammar and punctuation is critical to ensuring proper comprehension of the message being conveyed. In academic writing, adhering to these guidelines is particularly important in order to maintain a professional and credible tone. Therefore, writers should strive to write clearly and cogently while utilizing appropriate language and structure. The residual component, which denotes variation that can be attributed neither to seasonal nor trending components, is subsequently utilized for the identification of anomalies.

The fundamental concept underlying this methodology is premised on the notion that an anomaly is often typified by a considerable disparity between the residual component and the anticipated sample. Anomalies can be identified and signaled for further scrutiny by means of comparing factual balances with predetermined thresholds or employing statistical models. The proposed methodology exhibits diverse implementations in various domains, including but not limited to finance, industrial manufacturing, and healthcare. Particularly, this approach can be leveraged to detect anomalous trends within temporal data, which can signify fraudulent conduct, erroneous procedures, and other related phenomena. The utilization of equipment or the emergence of disease outbreaks.

3.2.4.2 Python library

For making abnormal detection, we have to know the form of our data. Our data is the vital signs data with time series so, we decided to use seasonal decomposition method to decompose data and use that data to make the abnormal detection. There are many Python libraries to import to our Jupyter notebook for running the abnormal detection code. In this detection, we use IPython, numpy, pandas, matplotli.pyplot, seaborn and warning.

```
from IPython.display import display

import numpy as np
import pandas as pd
pd.set_option('display.max_rows', 15)
pd.set_option('display.max_columns', 500)
pd.set_option('display.width', 1000)

import matplotlib.pyplot as plt
from datetime import datetime
from datetime import timedelta
from pandas.plotting import register_matplotlib_converters
from mpl_toolkits.mplot3d import Axes3D

import matplotlib.dates as mdates

from statsmodels.tsa.stattools import acf, pacf
from statsmodels.tsa.statespace.sarimax import SARIMAX
register_matplotlib_converters()
from time import time
import seaborn as sns
sns.set(style="whitegrid")

from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans
from sklearn.covariance import EllipticEnvelope

import warnings
warnings.filterwarnings('ignore')

RANDOM_SEED = np.random.seed(0)
```

Figure 61: Imported Python libraries to the abnormal detection.

3.2.4.3 Import dataset

After we imported Python libraries to our Jupyter notebook, we began with import our dataset (.csv) into our notebook.

Data

In [4]:

```
def parser(s):
    return datetime.strptime(s, '%m/%d/%Y')
```

In [5]:

```
#read data
vital_data = pd.read_csv('HR06-07.csv', parse_dates=[0], index_col=0, date_parser=parser)
```

In [6]:

```
#infer the frequency of the data
vital_data = vital_data.asfreq(pd.infer_freq(vital_data.index))
```

Figure 62: Imported dataset to the abnormal detection.

3.2.4.4 Manage dataset

Then after importing the dataset, we will process and reform our dataset to be an easy table. And that table was used to plot the graph in the next step.

Manage data

In [7]:
```
import datetime
idx = pd.IndexSlice
```

In [8]:
```
start_date = datetime.datetime(2022,6,1)
end_date = datetime.datetime(2022,7,31)
lim_vital_data = vital_data[start_date:end_date]
```

In [9]:
```
lim_vital_data
```

Out[9]:

	avgHR
Time	
2022-06-01	88
2022-06-02	85
2022-06-03	90
2022-06-04	91
2022-06-05	94
...	...
2022-07-27	95
2022-07-28	101
2022-07-29	96
2022-07-30	104
2022-07-31	97

61 rows × 1 columns

Figure 63: Managed dataset to be the table.

3.2.4.4 Plot graph

Next is a common process to use the table data plot into the graph. (Result will be in next chapter)

In [12]:
```
plt.figure(figsize=(12,5))
plt.axvspan(*mdates.datestr2num(['06/14/2022', '06/24/2022']), color='pink', alpha=0.5)
plt.plot(lim_vital_data)

plt.title('Average HeartRate June-July', fontsize=22)
plt.ylabel('HeartRate', fontsize=16)
plt.xlabel('Date', fontsize=16)
for year in range(start_date.year,end_date.year):
    plt.axvline(pd.to_datetime(str(year)+'-01-01'), color='k', linestyle='--', alpha=0.5)

plt.show()

```

Figure 64: Code for plotting average heart rate from June-July graph.

3.2.4.4 Seasonal Decomposition

For the average heart rate, we used that data to decompose as three parts of seasonal decomposition. They consist of trend, seasonality and residual. (Result will be in next chapter)

Seasonal Decompose

```
In [13]:
from statsmodels.tsa.seasonal import seasonal_decompose
import matplotlib.dates as mdates
```

```
In [14]:
plt.rc('figure',figsize=(12,8))
plt.rc('font',size=15)

result = seasonal_decompose(lim_vital_data,model='additive')
fig = result.plot()
#plt.axvspan(*mdates.datestr2num(['06/14/2022', '06/24/2022']), color='pink', alpha=0.5)
```

Figure 65: Code for making seasonal decomposition.

3.2.5 Artificial Intelligence (AI) System

3.2.5.1 AI model

For artificial intelligence model training, we use Python language and machine learning to create models. This model will create HROSAD. The artificial intelligence (AI) model shall undertake training of a logistic regression classifier on the training dataset, utilizing exclusively the normal instances, that is, those free from anomalies. Subsequently, the classifier will be assessed on the test set by utilizing the instances that did not undergo classification as anomalies. The assessment criteria employed consist of accuracy, precision, recall, and F1-score. These metrics will provide insight into the effectiveness of our model in identifying COVID-19 through the utilization of heart rate and step data.

3.2.5.2 Python library

In the first step, we need to import the library to create a machine-learning process. The library includes sys, argparse, pandas, Numpy, matplotlib.pyplot, matptotlib.dates, and seaborn. After that, we need to import the seasonal_decompose from statsmodel.tsa.seasonal library, StandardScaler from sklearn.preprocessing library, accuracy_score, confusion_matrix, f1_score, and recall_score from sklearn.metrics library.

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import IsolationForest
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
from sklearn.svm import OneClassSVM
```

Figure 66: Imported Python library to the machine learning model.

3.2.5.3 Import dataset

Detecting Covid-19 using abnormal heart rate and step values can be approached as a binary classification problem. We will train a machine learning model to classify whether a person is infected with Covid-19 based on their heart rate and step count data.

First, we need to prepare our data. Suppose we have data on the heart rate and steps of a group of people, and we also have information on whether they have been infected with Covid-19. We will use this data to train and test our model.

In [54]:
```

# Load data
data = pd.read_csv('HR_ST-CV06-12.csv')

# Split into features and target
X = data[['avgHR', 'steps']]
y = data['covid_positive']

# Split into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Standardize features
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

Figure 67: Imported dataset to the machine learning model.

3.2.5.4 Make the detector using Isolation Forest

Next, we will use the Isolation Forest algorithm to detect anomalies. This algorithm is based on the fact that the unusual cases are few and different from the normal cases. We will use this to detect abnormal heart rate and step values in our data.

In [55]:
```

# Fit the anomaly detector
detector = IsolationForest(n_estimators=100, contamination='auto', random_state=42)
detector.fit(X_train)

# Detect anomalies in the test set
y_pred = detector.predict(X_test)

# Map anomalies to binary labels
y_pred = (y_pred == -1).astype(int)
```

Figure 68: Made the detector with Isolation Forest for the machine learning model.

3.2.5.5 Training and get results.

Finally, we will use a machine learning classifier to classify our data based on detected anomalies. Here we will use a logistic regression classifier. (Result will be in next chapter)

In [60]:

```

# train the One-Class SVM model on the training set
clf = OneClassSVM(gamma='scale', nu=0.1)
clf.fit(X_train)

# predict the class labels of the test set
y_pred = clf.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred,average='macro')
recall = recall_score(y_test, y_pred,average='macro')
f1 = f1_score(y_test, y_pred,average='macro')


print('Accuracy: {:.2f}'.format(accuracy))
print('Precision: {:.2f}'.format(precision))
print('Recall: {:.2f}'.format(recall))
print('F1-score: {:.2f}'.format(f1))

```

Figure 69: The machine learning model training and results.

3.3 Interesting Problems

Since the Covid-19 pandemic, the spread of Coronavirus 2019 has caused turbulence to the medical system around the world. Medical staff would be the riskiest group to work in that situation because they were the first gate to face patients. And as it was a pandemic that can infect many people very easily. The result after many infected patients would be insufficient room, medical devices, and medical staff to take care of all patients.

In the current situation, the problem of congestion in the hospital is very serious due to there being new illnesses and viruses that can be transferred from person to person. And we don't know if it will have a new pandemic which comes from new viruses and can make damage like Covid-19 or more efficiently, we can't know.

3.4 Proposed Solution

From the problem that there are many patients from Covid-19 and this pandemic was easy to infect by the air so, we have the idea that we should have an early detection and alert system. But how can we detect it? We think that to detect diseases and pandemics, we have to use some data to analyze the symptoms and the data that can clearly show the difference between normal and abnormal conditions of our body is a vital sign. For the wearable which we can wear all the time and can measure vital signs all time is smartwatch. And this pandemic made everyone know about quarantine and as many patients made us know about home isolation. Both of those have difficult methods to send and receive vital signs data, telemedicine can make them faster and easier due to everything using the internet and Bluetooth to make real time connections. We combine all of those parts to be our project names “Pre-symptomatic detection of Covid-19 system based on deep learning technology from smartwatch data”.

3.5 Summary

The high-level specifications and design of a system with several essential elements for the data-gathering process were covered in this chapter. The measurer sensors for vital signs and important data like temperature, steps, and time are part of the smartwatch's original component. To bring out data from the smartwatches in each brand, has their own specific way like J-style has to make mobile application to get the vital signs data and Fitbit has more easy ways than. Fitbit have to access the web platform, then write Python code to retrieve data from Fitbit watch.. The application is secondary since telehealth requires a system for displaying vital signs, and the aim of this study is to create a telehealth system that is simpler and easier to use. The third is a patient database system connected to artificial intelligence (AI) to detect pre-symptomatic COVID-19 by examining patient symptoms filled out and vital signs obtained from the smartwatch.

CHAPTER 4

EXPERIMENTAL RESULT AND DISCUSSION

4.1 Introduction

From chapters 3 and 4 described the design and implementation of pre-symptomatic detection of Covid-19 system, a system that can detect Covid-19 pre-symptom using vital signs from smartwatches. In this chapter, we present a testing method and its results that consist of the results of Jstyle 2025E and the results of Fitbit and AI technology. For Jstyle 2025E model, its results include the Bluetooth Communication, our own application, and the data retrieval of Jstyle watches. For Fitbit change 5, its results include the API number communication, Fitbit application and the data retrieval of Fitbit sense. For the artificial Intelligence, it includes the graph result of pre-symptomatic Covid-19 and graph result of patients who were not infected Covid-19.

4.2 Result and Discussion

In the process of retrieving vital signs data, we have to wear smartwatches at all times to get accuracy and continuous value. Both smartwatches have their specific method to bringing out the data. For Jstyle, it should make the application with its API to connect to the Bluetooth in the watches for retrieving data out. In Fitbit platform, it can use Python code with token and client ID to request permission to bring out data from Fitbit's server.

4.2.1 Fitbit smartwatch

4.2.1.1 Heart rate

For heart rate data, the data that is retrieved show the value of heart rate per minute, value of resting heart rate, calories out, and data date. For heart rate data we export in the form of excel sheet (.xlsx) file per day and combine it to month data for training AI.

```
{'activities-heart': [{'dateTime': '2022-06-07', 'value': {'customHeartRateZones': [], 'heartR
0.79662, 'max': 123, 'min': 30, 'minutes': 1440, 'name': 'Out of Range'}, {'caloriesOut': 0, '
s': 0, 'name': 'Fat Burn'}, {'caloriesOut': 0, 'max': 180, 'min': 148, 'minutes': 0, 'name': '
ax': 220, 'min': 180, 'minutes': 0, 'name': 'Peak'}], 'restingHeartRate': 73}}], 'activities-h
[{'time': '00:00:00', 'value': 98}, {'time': '00:01:00', 'value': 98}, {'time': '00:02:00', 'v
0', 'value': 101}, {'time': '00:04:00', 'value': 102}, {'time': '00:05:00', 'value': 103}, {'t
{'time': '00:07:00', 'value': 87}, {'time': '00:08:00', 'value': 90}, {'time': '00:09:00', 'va
'value': 87}, {'time': '00:11:00', 'value': 92}, {'time': '00:12:00', 'value': 93}, {'time': '
e': '00:14:00', 'value': 106}, {'time': '00:15:00', 'value': 106}, {'time': '00:16:00', 'value
alue': 91}, {'time': '00:18:00', 'value': 99}, {'time': '00:19:00', 'value': 99}, {'time': '00
'00:21:00', 'value': 93}, {'time': '00:22:00', 'value': 94}, {'time': '00:23:00', 'value': 92}
87}, {'time': '00:25:00', 'value': 82}, {'time': '00:26:00', 'value': 84}, {'time': '00:27:00'
8:00', 'value': 89}, {'time': '00:29:00', 'value': 97}, {'time': '00:30:00', 'value': 100}, {'
```


Figure 70: The retrieval of heart rate data.

Datetime	HR
2022-06-16 00:00:00	104
2022-06-16 00:01:00	104
2022-06-16 00:02:00	103
2022-06-16 02:35:00	82
2022-06-16 02:36:00	70
2022-06-16 02:37:00	64
2022-06-16 02:38:00	73
2022-06-16 02:39:00	72
2022-06-16 02:40:00	68
2022-06-16 02:41:00	68
2022-06-16 02:42:00	62
2022-06-16 02:43:00	59
2022-06-16 02:44:00	58
2022-06-16 02:45:00	59
2022-06-16 02:46:00	60
2022-06-16 02:47:00	59
2022-06-16 02:48:00	57
2022-06-16 02:49:00	61
2022-06-16 02:50:00	59
2022-06-16 02:51:00	63
2022-06-16 02:52:00	66
2022-06-16 02:53:00	64

Table 2: Data of heart rate per day.

Time	avgHR
6/1/2022	88
6/2/2022	85
6/3/2022	90
6/4/2022	91
6/5/2022	94
6/6/2022	94
6/7/2022	95
6/8/2022	86
6/9/2022	85
6/10/2022	90
6/11/2022	80
6/12/2022	100
6/13/2022	84
6/14/2022	106
6/15/2022	97
6/16/2022	116
6/17/2022	108
6/18/2022	102
6/19/2022	102
6/20/2022	105
6/21/2022	103

Table 3: Data of heart rate per month.

4.2.1.2 Steps

In the context of this study, the metric of steps data refers to the dataset that displays the number of steps taken per minute, the resting heart rate, the amount of calories expended, and the date in which the data was gathered. The data obtained from the series of steps is subsequently exported as an excel sheet (.xlsx) file on a daily basis, and then aggregated into monthly data to facilitate the training of AI models.

```
<class 'dict'>
{'activities-steps': [{'dateTime': '2022-06-07', 'value': '13566'}], 'activities-steps-intraday': {'dataset': [{'time': '00:0
0:00', 'value': 0}, {'time': '00:01:00', 'value': 12}, {'time': '00:02:00', 'value': 20}, {'time': '00:03:00', 'value': 74},
{'time': '00:04:00', 'value': 24}, {'time': '00:05:00', 'value': 35}, {'time': '00:06:00', 'value': 14}, {'time': '00:07:00',
'value': 0}, {'time': '00:08:00', 'value': 10}, {'time': '00:09:00', 'value': 0}, {'time': '00:10:00', 'value': 0}, {'time':
'00:11:00', 'value': 0}, {'time': '00:12:00', 'value': 0}, {'time': '00:13:00', 'value': 51}, {'time': '00:14:00', 'value': 4
1}, {'time': '00:15:00', 'value': 13}, {'time': '00:16:00', 'value': 16}, {'time': '00:17:00', 'value': 0}, {'time': '00:18:0
0', 'value': 16}, {'time': '00:19:00', 'value': 21}, {'time': '00:20:00', 'value': 29}, {'time': '00:21:00', 'value': 4}, {'t
ime': '00:22:00', 'value': 5}, {'time': '00:23:00', 'value': 10}, {'time': '00:24:00', 'value': 0}, {'time': '00:25:00', 'val
ue': 0}, {'time': '00:26:00', 'value': 0}, {'time': '00:27:00', 'value': 0}, {'time': '00:28:00', 'value': 0}, {'time': '00:2
9:00', 'value': 0}, {'time': '00:30:00', 'value': 0}, {'time': '00:31:00', 'value': 0}, {'time': '00:32:00', 'value': 0}, {'t
ime': '00:33:00', 'value': 8}, {'time': '00:34:00', 'value': 44}, {'time': '00:35:00', 'value': 30}, {'time': '00:36:00', 'va
lue': 0}, {'time': '00:37:00', 'value': 0}, {'time': '00:38:00', 'value': 0}, {'time': '00:39:00', 'value': 0}, {'time': '00:
40:00', 'value': 0}, {'time': '00:41:00', 'value': 0}, {'time': '00:42:00', 'value': 0}, {'time': '00:43:00', 'value': 0},
{'time': '00:44:00', 'value': 0}, {'time': '00:45:00', 'value': 0}, {'time': '00:46:00', 'value': 0}, {'time': '00:47:00', 'v
alue': 0}, {'time': '00:48:00', 'value': 0}, {'time': '00:49:00', 'value': 0}, {'time': '00:50:00', 'value': 0}, {'time': '0
```

Figure 71: The retrieval of steps data.

Time	Steps
2022-07-01 10:52:00	16
2022-07-01 10:53:00	8
2022-07-01 10:54:00	41
2022-07-01 10:55:00	66
2022-07-01 10:56:00	97
2022-07-01 10:57:00	94
2022-07-01 10:58:00	101
2022-07-01 10:59:00	112
2022-07-01 11:00:00	109
2022-07-01 11:01:00	81
2022-07-01 11:02:00	35
2022-07-01 11:03:00	0

Table 4: Data of steps per day.

Time	steps
6/1/2022	1888
6/2/2022	905
6/3/2022	1924
6/4/2022	1252
6/5/2022	1410
6/6/2022	1897
6/7/2022	1099
6/8/2022	1643
6/9/2022	1381
6/10/2022	1581
6/11/2022	1670
6/12/2022	1336
6/13/2022	943
6/14/2022	582
[illegible]	[illegible]

Table 5: Data of steps per month.

4.2.2 J-style smartwatch

4.2.2.1 J-style java application

4.2.2.1.1 Bluetooth Low Energy Communication

After writing code to connect the Bluetooth Low Energy (BLE) communication by using the code to request the Bluetooth permission on the app's manifest file, the code to request the user's permission to enable Bluetooth, UUID of Bluetooth LE and the code to scan surrounding devices and show the list of devices name, the result displayed on the application is shown in Figure . To see the permission of Bluetooth connection.

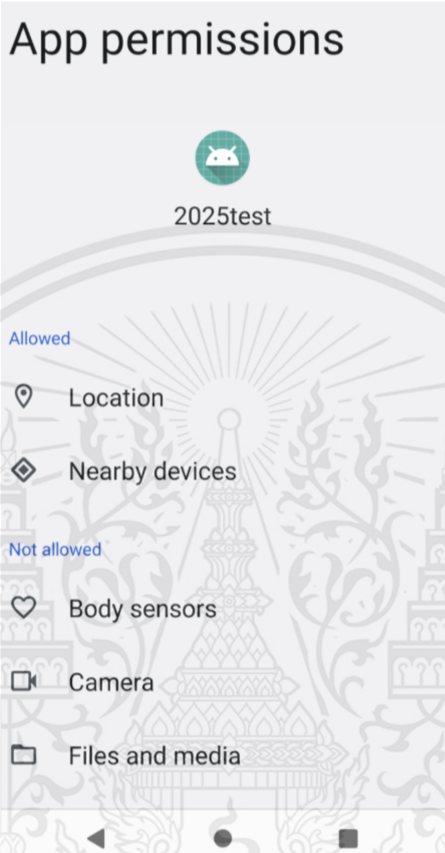


Figure 72: 2025E test permission.

4.2.2.1.2 All Functions and Real Time Monitor on Jstyle 2025E Application

For the main layout of this application that was developed, it is all functions and the real time monitor layout as shown in Figure that display all functions of this application that can set up the smartwatch and sync the data of the smartwatch. For the real time monitor, it displays the important data which is measured by the sensors of the Jstyle 2025E smartwatch in the real time such as steps, heart rate and temperature.

Figure 73: All functions of Jstyle 2025E application and the real time monitor.

4.2.2.1.3 Data retrieval of Jstyle Jstyle 2025E

4.2.2.1.3.1 Heart Rate

For this part, we retrieved data from the Jstyle 2025E smartwatch to display on the application by using the API number. In this model, we can retrieve the history of heart rate and step from our smartwatch.

For the heart rate data, we wrote the code to retrieve the history of heart rate data and then the user can click to read data on the HR and BP layout to display on the list of heart rate information as shown in Figure .

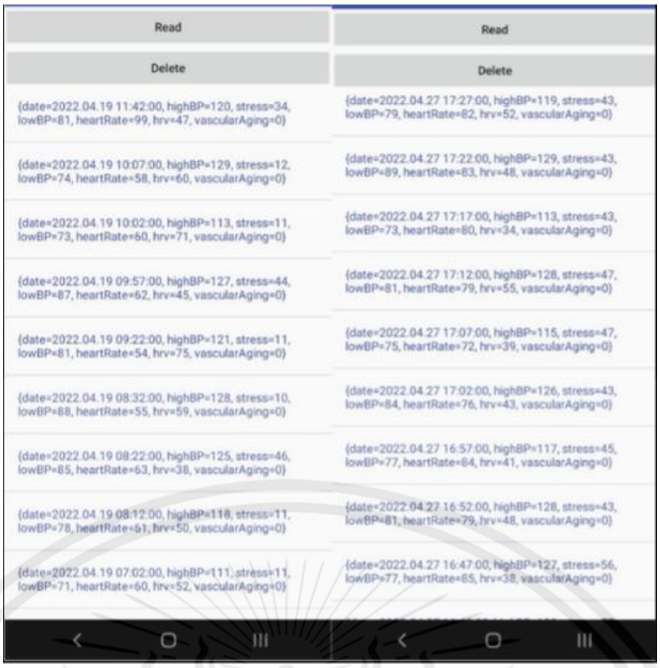


Figure 74: The list of the heart rate information on the application.

4.2.2.1.3.2 Steps

For the history of step data, our application can retrieve the step data from a smartwatch that measures the total step data per day. The user can see the history of step data as shown in figure.X by selecting a read button, and then the application will display the list of step information per day.

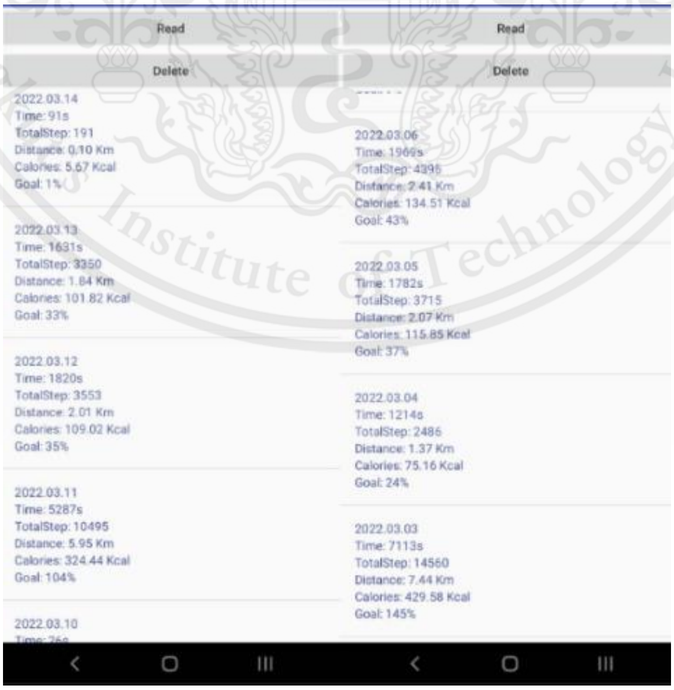


Figure 75: Total step data per day on the application

4.2.2.2 J-style kotlin application

4.2.2.2.1 Bluetooth Low Energy Communication

after installation Apps on mobile phones allow access to connect the Bluetooth Low Energy (BLE) communication by using the code to request the Bluetooth permission on the app's manifest file, the code to request the user's permission to enable Bluetooth, UUID of Bluetooth LE and the code to scan surrounding devices and show the list of devices name, the result displayed on the application is shown in Figure . To see the permission of Bluetooth connection.

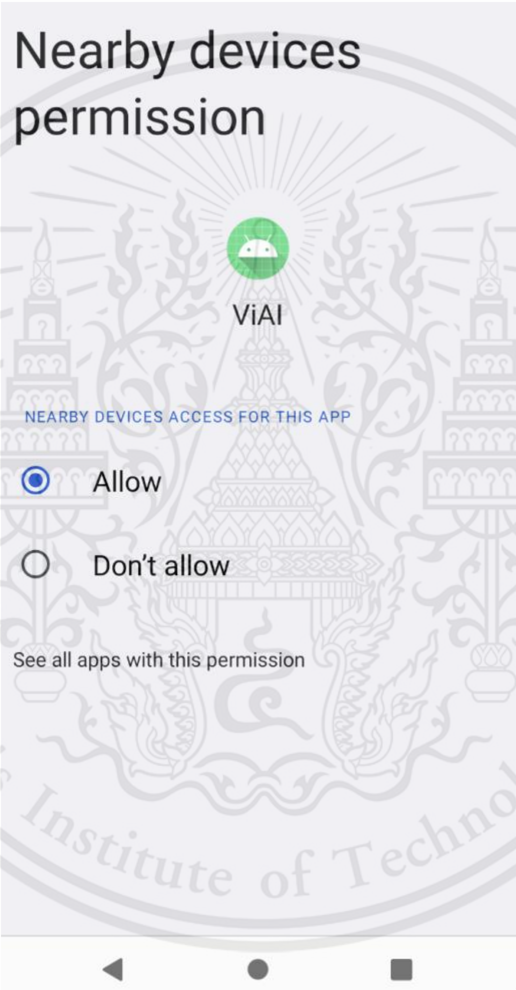


Figure 76: Nearby device permission.

4.2.2.2.2 login and registration system

When a user wants the to register for an application or service, they are typically asked to provide certain information to create an account. This information often includes a username and password, as well as other details like their email address or phone number then after user submits this information, the app or service will typically send it to a backend server, which may be powered by Firebase or another platform. The server will then process the information, verify that it meets certain criteria (such as having a unique username and a strong password), and store it in a database.

Once the user's information has been stored in the database, they will typically be able to log in to the app or service using their username and password. This may involve sending a request to the server to verify the user's credentials and retrieve any additional information associated with their account, such as their preferences or previous activity.

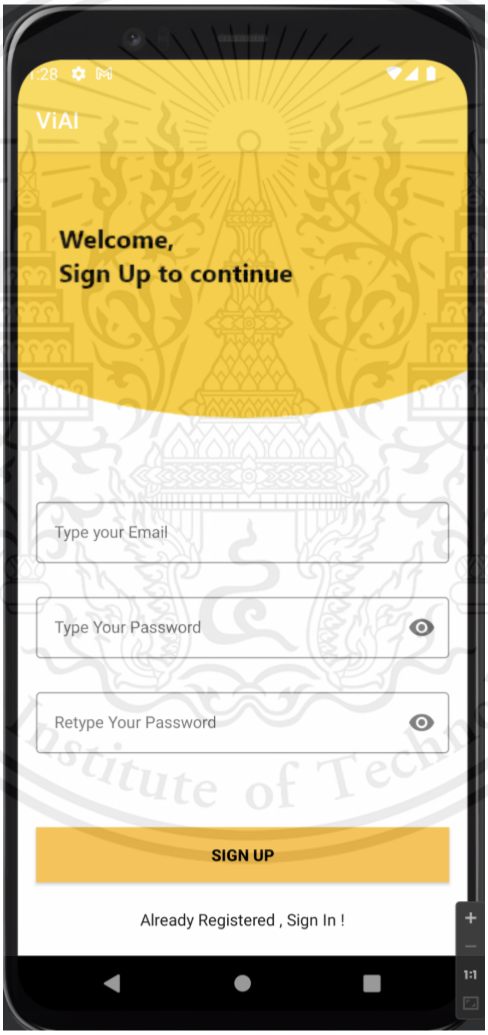


Figure 77: Sign up page.

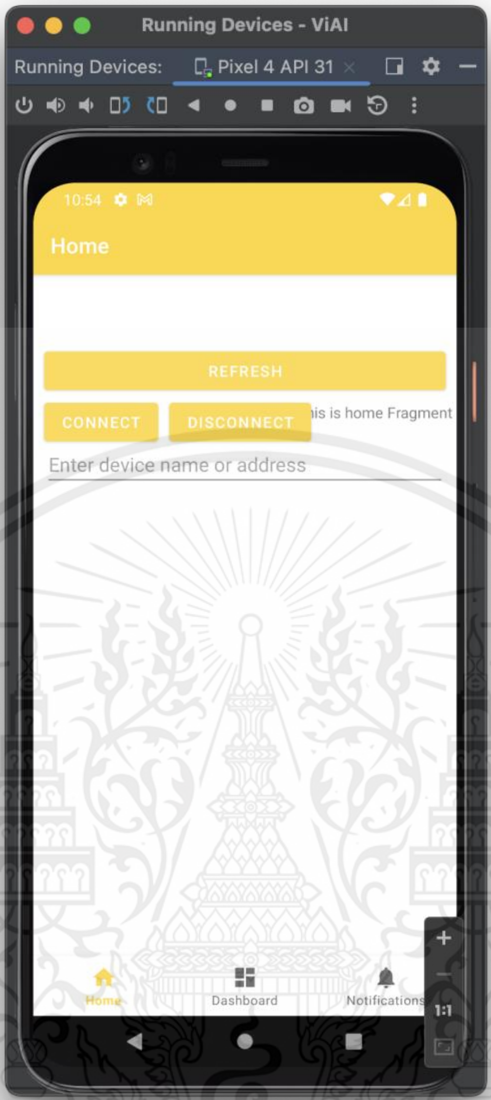


Figure 78: Connecting Bluetooth page.

4.2.3 web application

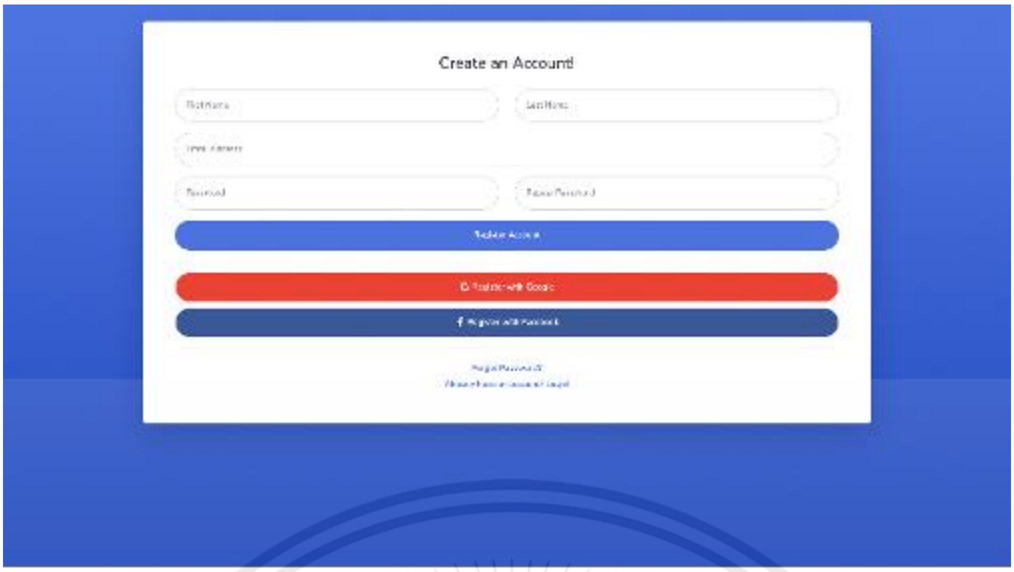


Figure 79: The creation of account page

Second, the patient and the doctor have to use their e-mail and password to log into our website. (Figure.16)

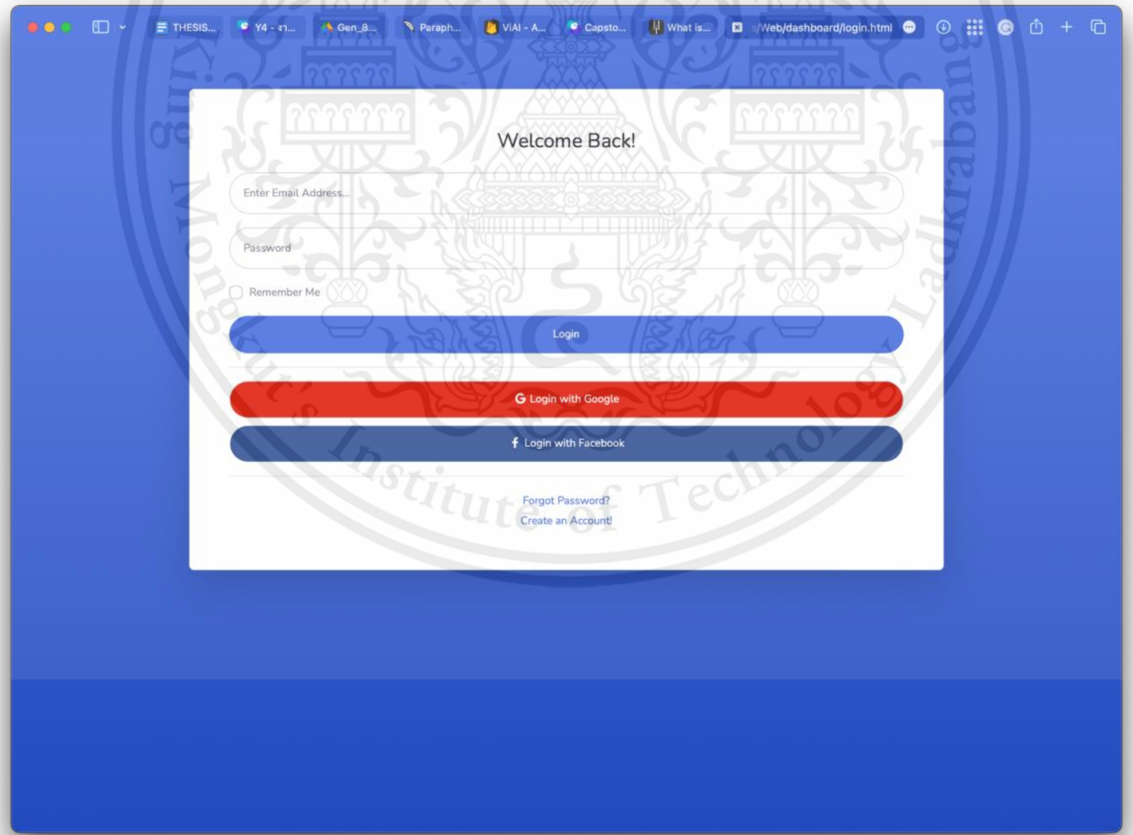


Figure 80: Log in page

Then, the patient and the doctor can see and monitor the information that we collected from smart watches in average values in one day. (Figure.17)

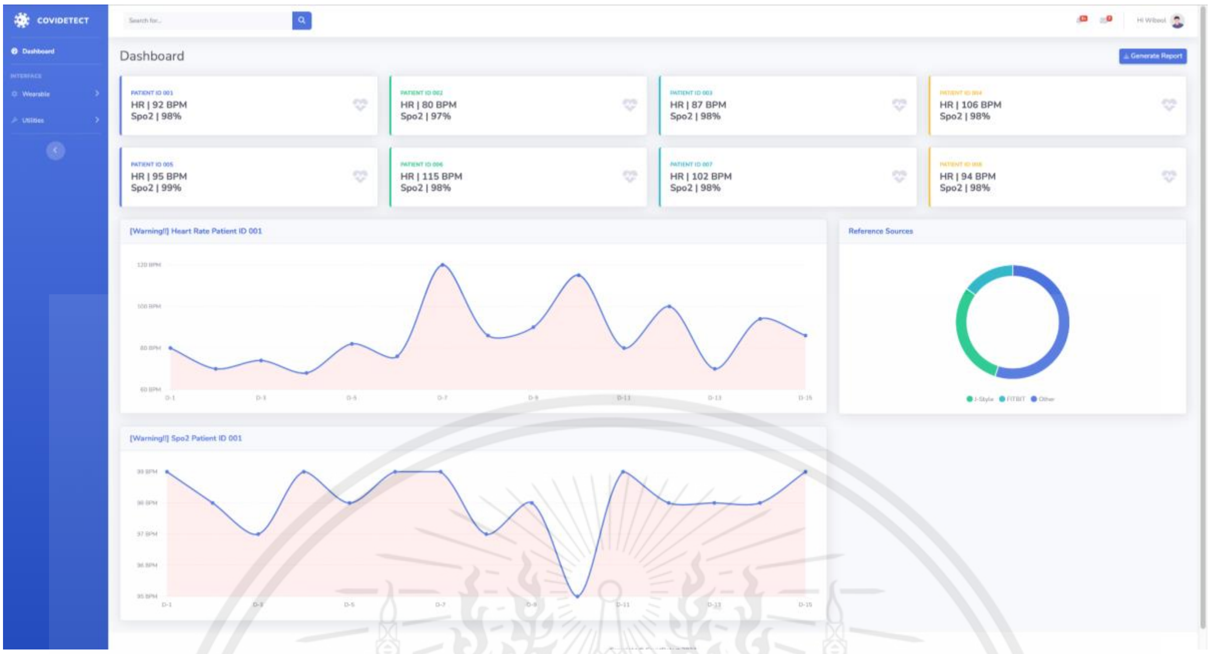


Figure 81: The monitor of the patient's page

In addition, We have 2 specific pages that can be show the data that have been received both bands for only Fitbit on Fitbit page (Figure.18) and J-style on J-style page. (Figure.19)

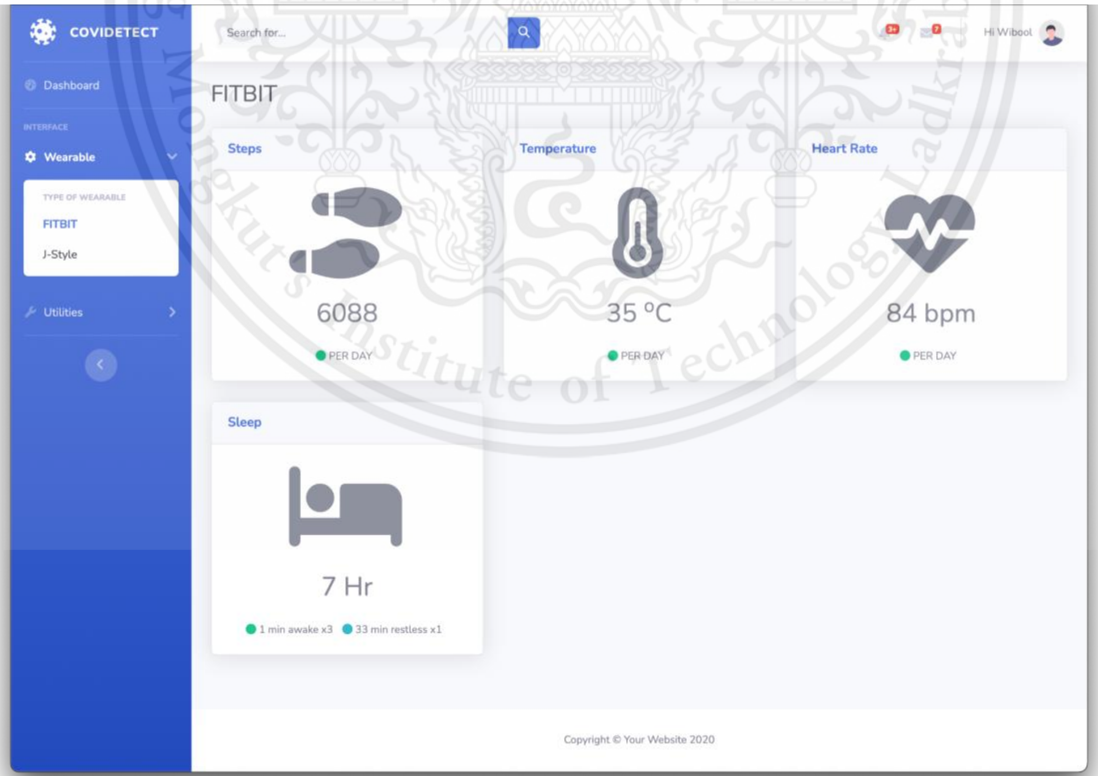


Figure 82: Fitbit's page

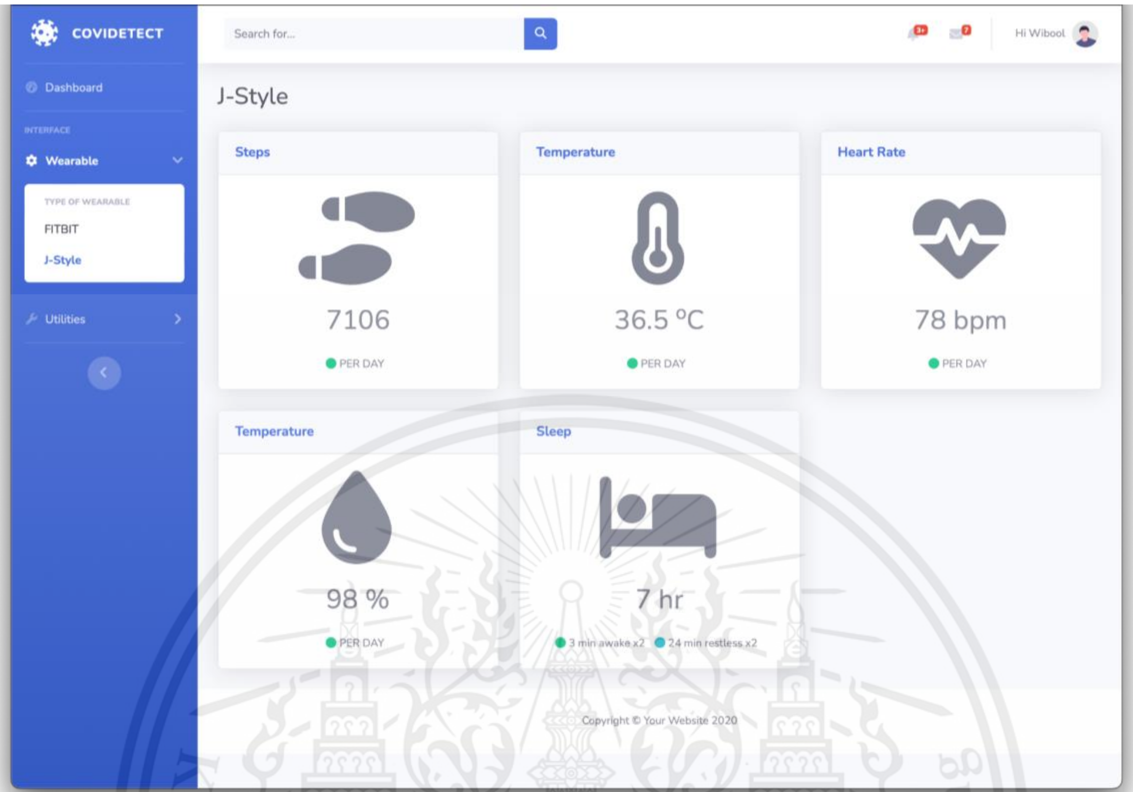


Figure 83: J-style's page

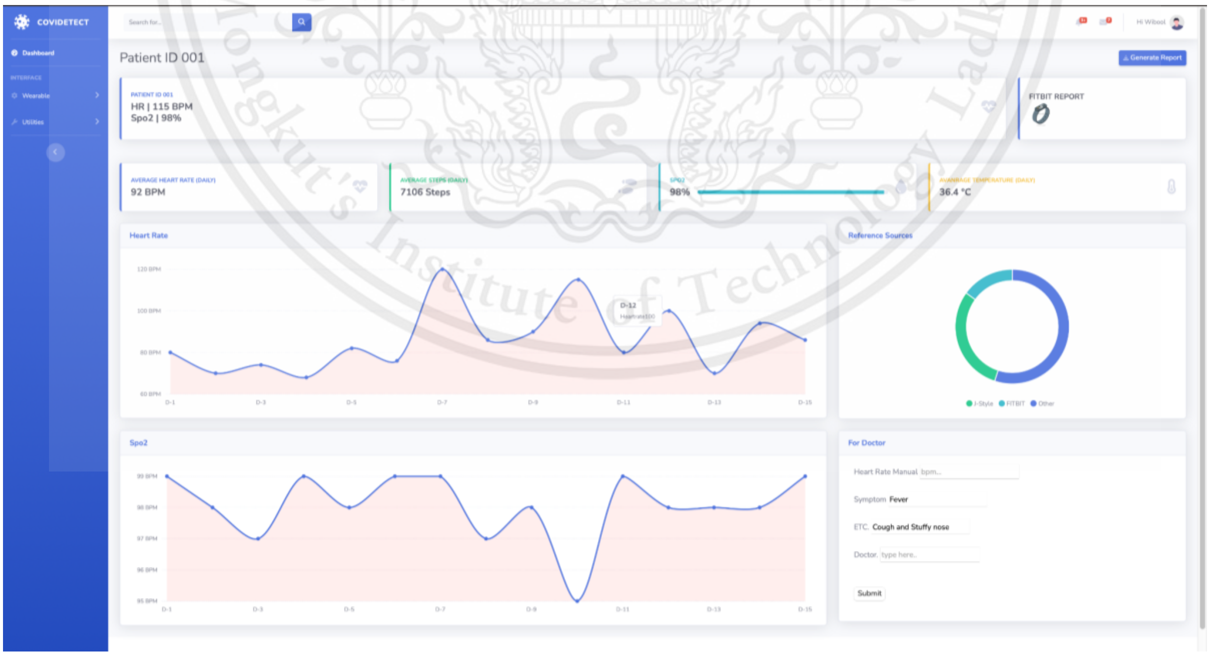


Figure 84: Patient # 01 page.

4.2.4 Abnormal detection and machine learning

4.2.3.1 Abnormal detection using seasonal decomposition

Before we decompose the data, we have the average heart rate from the June-July graph which we described about the code in the previous chapter.

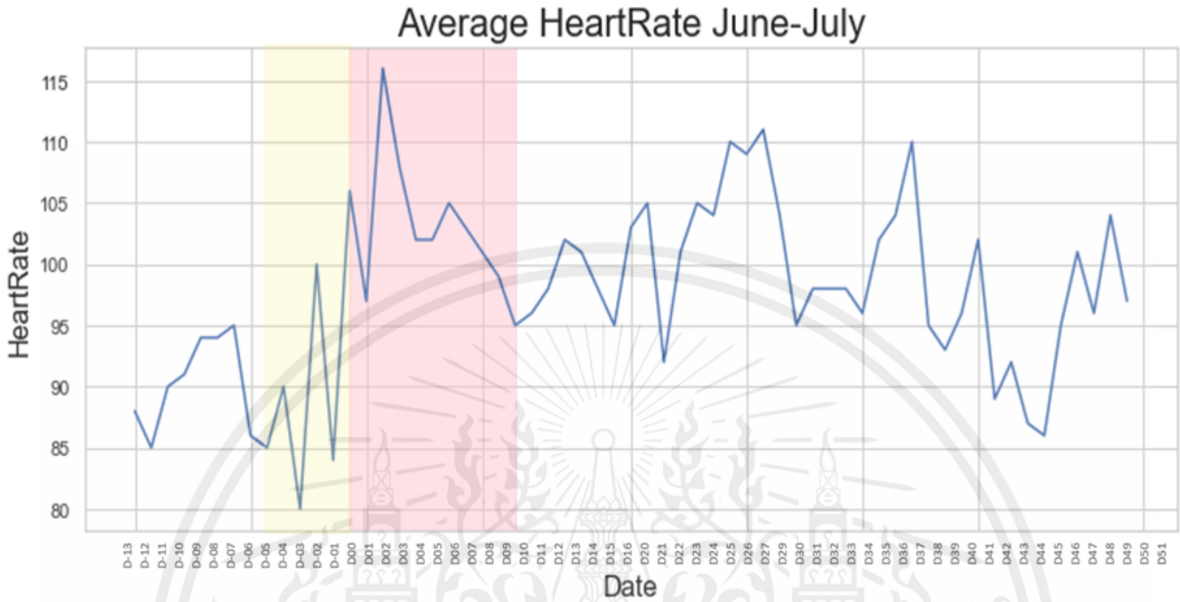


Figure 85: The average heart rate from June-July graph which red section is the real positive ATK test and the yellow section is the abnormal alert that risks being Covid-19.

Abnormal detection using seasonal decomposition is a technique used to identify and analyze anomalies or outliers in time-series data by decomposing the data into its seasonal, trend, and residual components.

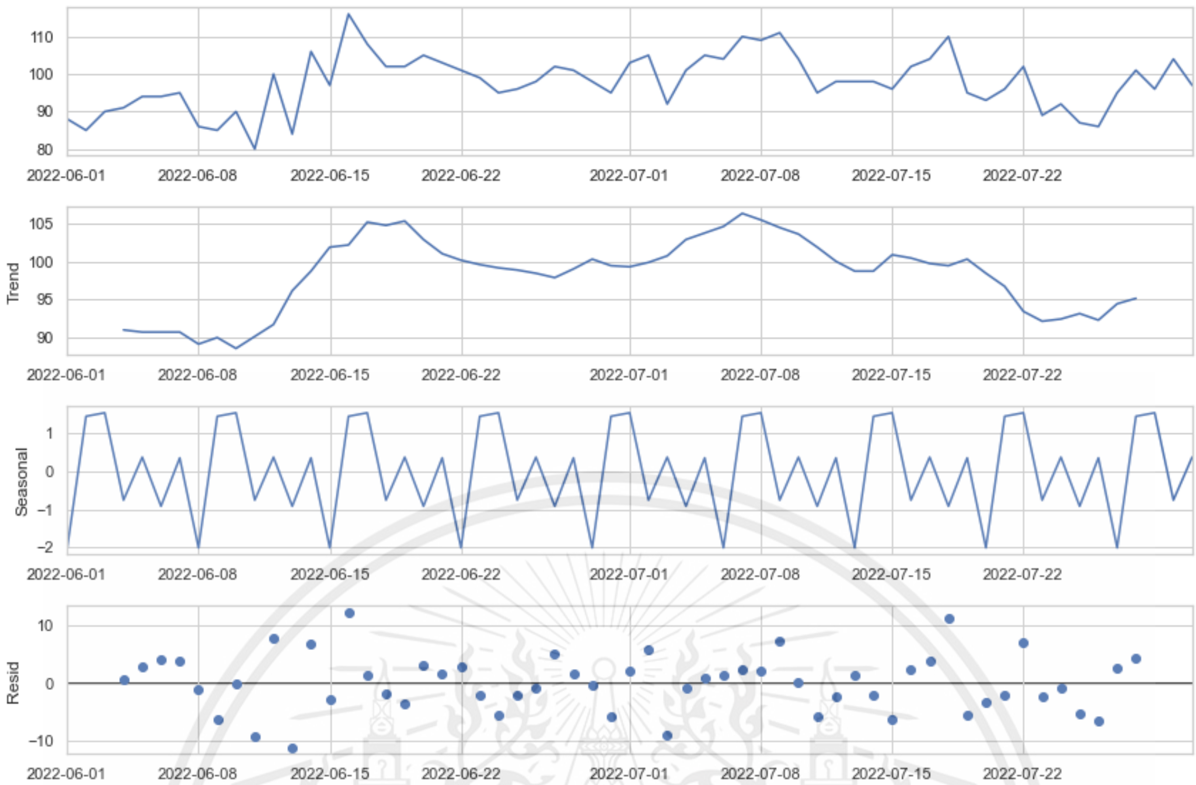


Figure 86: The data decomposition

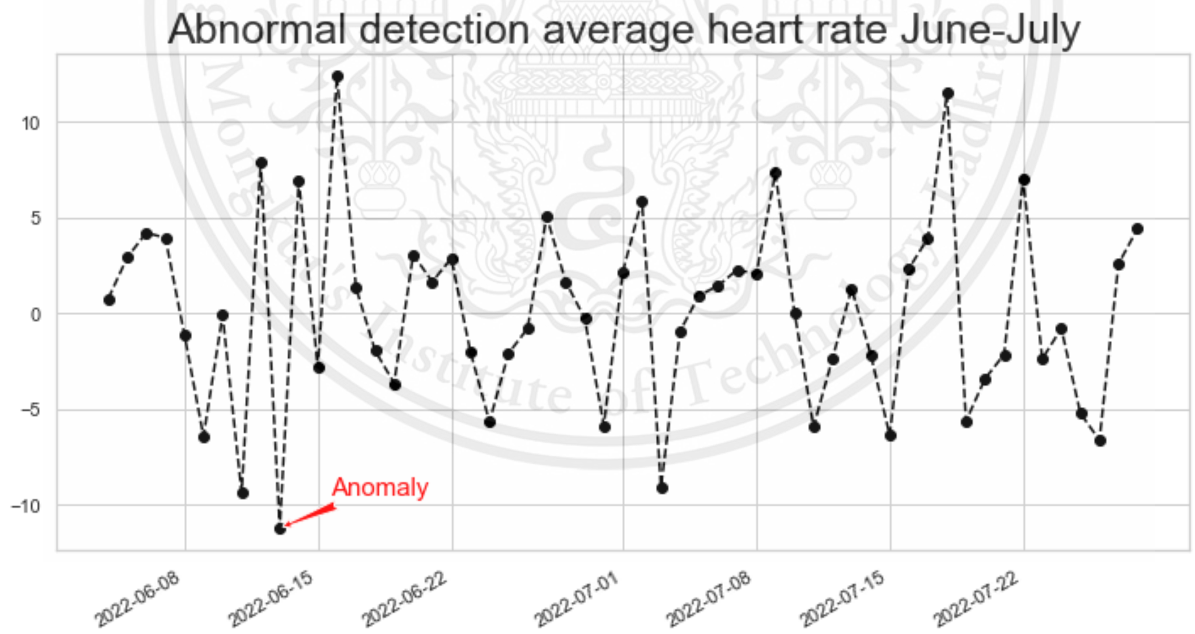


Figure 87: The residual value to detect abnormally value.

4.2.3.2 The machine learning and AI

As we have abnormal detection, we use machine learning for testing and training the model and our data. And the result got accuracy, precision, recall and F1 score.

```
Accuracy: 0.09
Precision: 0.04
Recall: 0.22
F1-score: 0.06
```

Figure 88: The result from the machine learning testing.

This set of results indicates the model's poor performance on the task. Here's a brief explanation of each metric:

Accuracy:
This metric measures the proportion of instances that are correctly classified among all instances in the test set. An accuracy score of 0.09 means that only 9% of cases were correctly classified by the model, which is a very low percentage.

Precision:
This metric measures the percentage of true positivity (cases correctly identified as positive) among all cases predicted by the model to be positive. A precision score of 0.01 indicates that only 1% of cases predicted by the model to be positive are actually true positives.

Recall:
This metric measures the rate of true positivity out of all cases that are truly positive. A recall score of 0.09 means that only 9% of positive cases were actually correctly identified by the model.

F1 score:
This metric is the harmonized mean of accuracy and recall. It combines two metrics to give a single score that represents the overall performance of the model. A low F1 score of 0.03 indicates that the model does not perform well on the task.

CHAPTER 5 CONCLUSION

5.1 Conclusion

This thesis expounds upon the development and execution of a pre-symptomatic detection mechanism intended for COVID-19. It is based on the utilization of vital signs information obtained from smartwatches to identify pre-symptomatic manifestations of COVID-19. This chapter elucidates a method for testing, comprising an evaluation of the outcomes attained by Jstyle 2025E and Fitbit smartwatches, alongside the results garnered from the utilization of AI technology.

The findings indicate that each smartwatch possesses a distinct approach for obtaining crucial physiological information, which is then extracted as an excel spreadsheet in the .xlsx format on a daily basis. These data sets are subsequently compiled and combined into monthly statistics to train artificial intelligence models. In accordance with Jstyle 2025E and Fitbit, data acquisition techniques differ. In the case of Jstyle 2025E, a proprietary application and API are employed to connect with the watches' Bluetooth functionality and retrieve data. In contrast, for Fitbit, Python code, along with a token and client ID, is utilized to solicit authorization to extract data from Fitbit's servers.

The present study involves the development of the Jstyle 2025E application, which has the capacity to selectively extract heart rate and step data chronicles from smartwatch devices. The application enables users to visualize the historical trends of these vital health metrics through a user-friendly interface. Moreover, the J-style Kotlin application developed in the present study incorporates a login and registration mechanism, which affords users the ability to establish an account with a username, password, email address, or phone number.

In this study, a pre-symptomatic detection system for COVID-19 utilizing smartwatches has been developed and demonstrated to possess promising capabilities in detecting pre-symptomatic signs of COVID-19. The insights presented in this thesis have important implications for future researchers and developers to enhance the system's functions and performance.

5.2 Discussion

As our project had to retrieve the vital signs data from smartwatches, Fitbit is easier than the J-style that it can bring out from Fitbit dev web platform but the J-style is different. It had to make the mobile application to encode and bring out vital signs data out to the computer. Unfortunately, the Android Gradle did not fit the updated version. Because Google always updates Gradle in the several times, that made us have to remake the new one because Gradle is A build automation tool that serves to automate the process of application creation.

The process of constructing a programmed application encompasses the procedures of compiling, linking, and packaging the software code. The utilization of build automation tools facilitates an increased level of consistency within the process.

Our project aims to build deep learning technology or machine learning to make the pre-symptomatic detection of Covid-19 system but our model accuracy is only 0.09 or 9%. It was very low because the main reason was that our module and other components in the machine learning model was not suitable for the data and detection type. On the other hand, the another main reason is the very small number of dataset. Due to the J-style has to the mobile application and that application had problem so, the charge to get data from J-style watch because we had finished it just only the previous week and most of the smartwatches in our lab are J-style smartwatch. Because they are cheaper than Fitbit charge 5 that have only 3 watches and had only one data that infected COVID-19 even though we tried to find other vital signs dataset from an outsource but it could not be used in our model.

REFERENCES

[1] "Coronavirus," World Health Organization. [Online]. Available: https://www.who.int/health-topics/coronavirus. [Accessed: 20-Apr-2023].

[2] Z. Lv and Y. Li, "Wearable sensors for vital signs measurement: A survey," MDPI, https://www.mdpi.com/2224-2708/11/1/19 (Accessed: 22-Apr-2023).

[3] Huang C;Wang Y;Li X;Ren L;Zhao J;Hu Y;Zhang L;Fan G;Xu J;Gu X;Cheng Z;Yu T;Xia J;Wei Y;Wu W;Xie X;Yin W;Li H;Liu M;Xiao Y;Gao H;Guo L;Xie J;Wang G;Jiang R;Gao Z;Jin Q;Wang J;Cao B; "Clinical features of patients infected with 2019 novel coronavirus in Wuhan, China," Lancet (London, England). [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/31986264/. [Accessed: 25-Apr-2023].

[4] E. Monaghesh and A. Hajizadeh, "The role of telehealth during COVID-19 outbreak: A systematic review based on current evidence - BMC public health," BioMed Central, 01-Aug-2020. [Online]. Available: https://bmcpublichealth.biomedcentral.com/articles/10.1186/s12889-020-09301-4. [Accessed: 25-Apr-2023].

[5] W. Jiang, S. Majumder, S. Kumar, S. Subramaniam, X. Li, R. Khedri, T. Mondal, M. Abolghasemian, I. Satia, and M. J. Deen, "A wearable tele-health system towards monitoring COVID-19 and chronic diseases," IEEE Reviews in Biomedical Engineering, vol. 15, pp. 61–84, 2022.

[6] B. Scholkopf, "Support Vector Machines and kernel algorithms." [Online]. Available: https://alex.smola.org/papers/2002/SchSmo02b.pdf. [Accessed: 15-Apr-2023].

[7] A. Sidharan, "Support Vector Machine algorithm," GeeksforGeeks, 31-Mar-2023. [Online]. Available: https://www.geeksforgeeks.org/support-vector-machine-algorithm/. [Accessed: 25-Apr-2023].

[8] A. O. Shalash, A. Azuar, H. Y. R. Madge, N. Modhiran, A. A. Amarilla, B. Liang, A. A. Khromykh, D. Watterson, P. R. Young, I. Toth, and M. Skwarczynski, "Detection and quantification of SARS-COV-2 receptor binding domain neutralization by a sensitive competitive Elisa Assay," Vaccines, 16-Dec-2021. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8705285/. [Accessed: 25-

Apr-2023]

[9] P. Pongchairerks and J. Putponpai, “Deterministic, stochastic and local search algorithms for real-world ATSP instances,” International Journal of Logistics Systems and Management, vol. 24, no. 2, p. 265, 2016.

[10] A. C. Smith, E. Thomas, C. L. Snoswell, H. Haydon, A. Mehrotra, J. Clemensen, and L. J. Caffery, “Telehealth for global emergencies: Implications for coronavirus disease 2019 (covid-19),” Journal of Telemedicine and Telecare, vol. 26, no. 5, pp. 309–313, 2020.

[11] P. Pongchairerks and J. Putponpai, “Deterministic, stochastic and local search algorithms for real-world ATSP instances,” International Journal of Logistics Systems and Management, vol. 24, no. 2, p. 265, 2016.

[12] S. Cheriyedath, “Photoplethysmography (PPG),” News, 27-Feb-2019. [Online]. Available: https://www.news-medical.net/health/Photoplethysmography-(PPG).aspx. [Accessed: 25-Apr-2023].

[13] M. Kunz, “Recommended steps per day by age: A complete guide,” Runstreet, https://www.runstreet.com/blog/recommended-steps-per-day-by-age [Accessed: 21-Apr-2023].

[14] K. Chouhbi, “Preprocessing data: Feature scaling,” Medium, 08-Jul-2020. [Online]. Available: https://towardsdatascience.com/preprocessing-data-feature-scaling-cc28c508e8af?gi=484f0f2af018. [Accessed: 25-Apr-2023].

[15] K. Lerdlamyong, “SAR-CoV-2 specific antibody detection among heath care workers with direct contact to COVID-19 patients in Vachira Phuket Hospital,” Region 11 Medical Journal, https://he02-old.tci-thaijo.org/index.php/Reg11MedJ/article/view/248985 (accessed Jun. 5, 2023).

[16] P. Pongchairerks and J. Putponpai, “Deterministic, stochastic and local search algorithms for real-world ATSP instances,” International Journal of Logistics Systems and Management, vol. 24, no. 2, p. 265, 2016.

[17] C. M. Lee, S.-P. Jin, E. J. Doh, D. H. Lee, and J. H. Chung, “Regional variation of human skin surface temperature,” Annals of dermatology, https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7992731/ [Accessed: 25-Apr-2023].

[18] “Normal oxygen level,” Nonin, https://www.nonin.com/resource/what-is-your-normal-oxygen-level/ [Accessed: 25-Apr-2023].

[19] S. Cheriyedath, "Photoplethysmography (PPG)," News, 27-Feb-2019. [Online]. Available: https://www.news-medical.net/health/Photoplethysmography-(PPG).aspx. [Accessed: 25-Apr-2023].

[20] C. C. medical professional, "Coronavirus disease (COVID-19): Symptoms & causes," Cleveland Clinic, https://my.clevelandclinic.org/health/diseases/21214-coronavirus-covid-19 (accessed Jun. 5, 2023).

[21] "SARS-COV-2 variant classifications and definitions," Centers for Disease Control and Prevention. [Online]. Available: https://www.cdc.gov/coronavirus/2019-ncov/variants/variant-classifications.html. [Accessed: 25-Apr-2023].

[22] "SARS-COV-2 variant classifications and definitions," Centers for Disease Control and Prevention. [Online]. Available: https://www.cdc.gov/coronavirus/2019-ncov/variants/variant-

[23] T. Kanstrén, "A look at precision, recall, and F1-score," Medium, 19-May-2021. [Online]. Available: https://towardsdatascience.com/a-look-at-precision-recall-and-f1-score-36b5fd0dd3ec. [Accessed: 25-Apr-2023].

[24] A. O. Shalash, A. Azuar, H. Y. R. Madge, N. Modhiran, A. A. Amarilla, B. Liang, A. A. Khromykh, D. Watterson, P. R. Young, I. Toth, and M. Skwarczynski, "Detection and quantification of SARS-COV-2 receptor binding domain neutralization by a sensitive competitive Elisa Assay," Vaccines, 16-Dec-2021. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8705285/. [Accessed: 25-Apr-2023].

[25] "Vital signs: How to check my vitals at home," Cleveland Clinic. [Online]. Available: https://my.clevelandclinic.org/health/articles/10881-vital-signs. [Accessed: 25-Apr-2023].

[26] W. Jiang, S. Majumder, S. Kumar, S. Subramaniam, X. Li, R. Khedri, T. Mondal, M. Abolghasemian, I. Satia, and M. J. Deen, "A wearable tele-health system towards monitoring COVID-19 and chronic diseases," IEEE Reviews in Biomedical Engineering, vol. 15, pp. 61–84, 2022.

[27] Y. Peddireddi, "Dealing with anomalies in the data," Analytics Vidhya, 28-Apr-2021. [Online]. Available: https://www.analyticsvidhya.com/blog/2021/04/dealing-with-anomalies-in-

the-data/. [Accessed: 25-Apr-2023].

[28] U. Yokkampon, S. Chumkamon, A. Mowshowitz, R. Fujisawa, and E. Hayashi, “Anomaly detection using support vector machines for time series data,” Journal of Robotics, Networking and Artificial Life, 31-May-2021. [Online]. Available: https://www.atlantis-press.com/journals/jrnal/125957120/view. [Accessed: 25-Apr-2023].

[29] A. Kharwal, “Standardscaler in machine learning: Aman Kharwal,” thecleverprogrammer, 24-Jun-2021. [Online]. Available: https://thecleverprogrammer.com/2020/09/22/standardscaler-in-machine-learning/. [Accessed: 25-Apr-2023].

[30] P. Serafeim Loukas, “How scikit-learn’s standardscaler works,” Medium, https://towardsdatascience.com/how-and-why-to-standardize-your-data-996926c2c832 (Accessed: 12-Mar-2023).

[31] P F. T. Liu, K. M. Ting and Z. -H. Zhou, "Isolation Forest," 2008 Eighth IEEE International Conference on Data Mining, Pisa, Italy, 2008, pp. 413-422, doi: 10.1109/ICDM.2008.17.

[32] D. Franco, “A visual guide to time series Decomposition analysis,” Encora, https://www.encora.com/insights/a-visual-guide-to-time-series-decomposition-analysis (Accessed: 10-Mar-2023).

[33] “Smartwatch with ECG and blood pressure,” Shenzhen Youhong Technology Co., Ltd., https://www.jointcorp.com/smart-watch/2025E-Smart-ECG-Watch.html (Accessed: 5-Feb-2023).

[34] “Fitbit Charge 5,” Advanced fitness + health tracker | Shop Fitbit Charge 5, https://www.fitbit.com/global/th/products/trackers/charge5 (Accessed: 5-Feb-2023).

[35] “Library in python - javatpoint,” www.javatpoint.com, https://www.javatpoint.com/library-in-python (Accessed: 7-Feb-2023).

[36] “Overview: The dart language,” Dart. [Online]. Available: https://dart.dev/guides/language. [Accessed: 25-Apr-2023].

[37] “Flutter app development - why pick flutter for Digital Products,” Nomtek, https://www.nomtek.com/blog/flutter-development (Accessed: 24-Apr-2023).

[38] “Android Studio,” Processing for Android, https://android.processing.org/tutorials/android_studio/index.html (Accessed: 7-Feb-2023).

[39] Fran, “Java App,” FutureLearn, https://www.futurelearn.com/info/blog/what-is-java-used-for (Accessed: 7-Feb-2023).

[40] C. Team, “What is Kotlin used for?,” Codecademy Blog, https://www.codecademy.com/resources/blog/what-is-kotlin-used-for/ (Accessed: 7-Feb-2023).

[41] “Firebase - the powerful backend-as-a-service framework,” devedis, https://devedis.ch/en/technologies/firebase (Accessed: 8-Feb-2023).