Project No. 4

YellowVet - Smart Veterinary in Your Pocket

Presented by

1. Mr. Noppawut Pukpinyo	58070503414
2. Mr. Pongnut Jittipanyakul	58070503419
3. Mr. Tanet Sriamorn	58070503452

Advisor

Assoc. Prof. Peerapon Siripongwutikorn, Ph.D.

"I've read and approved the content of this report"

…….

(…….)

Advisor

YellowVet - Smart Veterinary in Your Pocket

Mr. Noppawut Pukpinyo
Mr. Pongnut Jittipanyakul
Mr. Tanet Sriamorn

A Project Submitted in Partial Fulfillment of the Requirements
for the Degree of Bachelor of Engineering
Department of Computer Engineering, Faculty of Engineering
King Mongkut's University of Technology Thonburi
Academic Year 2018

YellowVet - Smart Veterinary in Your Pocket

1. Mr. Noppawut Pukpinyo 58070503414
2. Mr. Pongnut Jittipanyakul 58070503419
3. Mr. Tanet Sriamorn 58070503452

A Project Submitted in Partial Fulfillment of the Requirements
for the Degree of Bachelor of Engineering
Department of Computer Engineering, Faculty of Engineering
King Mongkut's University of Technology Thonburi
Academic Year 2018

Project Committee

…… Committee
(Lecturer Sally E. Goldin, Ph.D.)

…… Committee
(Asst. Prof. Nuttarnat Facundes, Ph.D.)

…… Committee
(Asst. Prof. Marong Phadoongsidhi, Ph.D.)

Project Title	YellowVet - Smart Veterinary in Your Pocket
Project Credit	3 credits
Project Participant	Mr. Noppawut Pukpinyo
	Mr. Pongnut Jittipanyakul
	Mr. Tanet Sriamorn
Advisor	Assoc. Prof. Peerapon Siripongwutikorn, Ph.D.
Degree of Study	Bachelor's Degree
Department	Computer Engineering
Academic Year	2018

Abstract

The abstract of this project is about understanding animal case studies in question and answer terms including observe an animal behaviour to make an application which work like a real veterinarian who can diagnose to many abnormal symptoms of pets. We mainly use Kotlin language to build our application and also have machine learning model that learned from many past animal case studies. We stored this predictive model on Realtime database which is Firebase from Google. Moreover, we use IoT device technology which can track pet’s behavior and helps diagnosis system.

From making this project, we evaluate a performance of our application by using diagnosis score which is similar to accuracy indicator of our veterinarian simulator. In this project our application is designed for using with dog only, because it is the most popular pet in Thailand. In future, we may make feature for other pet.

หัวข้อโครงงาน	เยลโลว์เว็ท - สัตวแพทย์อัจฉริยะในกระเป๋าคุณ
หน่วยกิตของโครงงาน	3 หน่วยกิต
จัดทำโดย	นายนพวุฒิ พุกภิญโญ
	นายปองณัฐ จิตติปัญญากุล
	นายธเนศ ศรีอมร
อาจารย์ที่ปรึกษา	รองศาสตราจารย์ ดร. พีรพล ศิริพงศ์วุฒิกร
ระดับการศึกษา	วิศวกรรมศาสตรบัณฑิต
ภาควิชา	วิศวกรรมคอมพิวเตอร์
ปีการศึกษา	2561

บทคัดย่อ

โครงงานนี้ได้ทำการศึกษาเกี่ยวกับเคสการรักษาของสัตว์เลี้ยงต่างๆในรูปแบบคำถาม และ คำตอบ รวมไปถึง การตรวจสอบพฤติกรรมของสัตว์เลี้ยง และสร้างแอพพลิเคชั้น ที่สามารถทำหน้าที่เสมือนสัตวแพทย์จำลองซึ่งสามารถวินิจฉัยอาการผิดปกติ
ของสัตว์เลี้ยงผู้ใช้ได้ โดยใช้ภาษาKotlinเป็นปัจจัยหลักในการสร้างแอพพลิเคชั้น รวมถึงมีโมเดลการเรียนรู้ (Machine Learning model) ที่สามารถวินิจฉัย
จากการเรียนรู้เคสการรักษา ถูกเก็บไว้บนระบบ Realtime database อย่าง Firebase ของ Google ซึ่งสามารถสื่อสารกับแอพพลิเคชั่น รวมไปถึงมีอุปกรณ์ IoT ที่สามารถติดตามพฤติกรรมสัตว์เลี้ยง และช่วยในการวินิฉัยอาการผิดปกติต่างๆ

จากการพัฒนาแอพพลิเคชันดังกล่าวได้ทำการประเมินประสิทธิภาพของการวินิฉัยด้วย diagnosis score ที่ทำหน้าที่เหมือน ตัววัดความแม่นยำของ สัตว์แพทย์จำลอง ในโปรเจคนี้แอพพลิเคชั่นของเราถูกออกแบบมาให้ใช้กับสุนัขเท่านั้น เนื่องจากเป็น
สัตว์เลี้ยงที่ค่อนข้างนิยมในประเทศไทย และ จะเพิ่มเติมสัตว์เลี้ยงชนิดอื่นๆในภายหลัง

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Problem Statement and Approach

In past few years ago, pet business market has grown tremendously. According to information from N.C.C. Exhibition Organizer Co., Ltd. (NEO) as organizer of Pet Expo Thailand 2018 [1], overview of pet business market has grown around 10-15% every year. Also in 2018, they said that pet business market will grow at least 10% or around 32 billion baht increase from 29 billion baht from year 2017. This shows that pets are very popular nowadays and make pet cheap and easier to own. However, many owners are still lack of readiness for pet ownership including money, time and knowledge. Lacking of these things may harm to their pet from late treatment problem.

Our group realize how important of problem about lack of readiness to take care of pets. We decide to create application which can reduce the burden of pet owner by applying performance of machine learning techniques for creating veterinarian simulator which can answer about what an anomaly happened to their pet. Moreover, we also provide advice or information about anomaly or symptom to reduce burden of pet owner and prevent dangerous from receiving late treatment.

1.2 Objectives

1. To develop an application that can warn pet owners if their pet has a problem.
2. To reduce the injury or death of pets resulting from lack of knowledge in caring for pets.
3. To apply knowledge and technology to facilitate the People with pets.

1.3 Scope

Scope of this project is creating veterinary simulator that can predict or answer the illness of pet from information which user fill as input.

By our AI veterinary system. The scope is as follows:

- The system will receive the pet's information from the user in form of text only. The user can enter that information in our application.
- Machine learning model to create illness classification from information that users provide.
- After system get data from user, Application will show a result of predict by infographics to easily to understand about risks, opportunities, and guide information pets.
- The results will come in the form of probabilities. Our veterinarian simulator will learn from the information of treatment cases from data of the pet hospital and question- answering veterinarian webboard.
- Hardware inspects pet behavior for use in pet detection.
    - Step counting for measuring movement behavioral of user's pet.
    - Breathing sensor for measuring breathing behavioral of user's pet.

1.4 Tasks and Schedule

1. Contact veterinarian and finding a data source.
2. Research IoT about pet behavioral tracking equipment.
3. Research about how to apply deep learning for creating veterinarian simulator.
4. Learning Android Studio and Kotlin language for building android application.
5. Develop a simple tracking equipment for pet.
6. Develop prototype veterinarian simulator.
7. Develop prototype YellowVet application wireframe.
8. Integrating IoT device and Machine learning model to application.
9. Application verification and validation.
10. Test with users and get their feedback about tracking equipment, prototype of veterinarian simulator.
11. Deploy application to Play Store.
12. Conclude the project.

Table 1.1 Gantt chart of YellowVet

Term 1

Activity / Week	September				October				November				December			
Contact veterinary and finding a data source.																
Research IoT about pet behavioral tracking equipment.																
Research about how to apply deep learning for creating veterinary simulator.																
Learning Android Studio and Kotlin language for building application.																
Develop a simple tracking equipment for pet.																
Develop prototype veterinary simulator.																
Develop prototype YellowVet application wireframe.																

Term 2

Activity / Week	January				February				March				April				May			
Reconstruct our report.																				
Train model and export to Firebase.																				
Machine learning model to application.																				
Test breathing sensor.																				
Connect IoT device to application.																				
Integrating IoT device to application.																				
Application verification and validation.																				
Test with users and get their feedback.																				
Deploy application to Play Store.																				
Conclude the project.																				

Color: **Green** = All, **Red** = Noppawut, **Orange** = Tanet, **Blue** = Pongnut

The table above shows the tasks that we need to complete for both terms. At the end of the first term, we will finish prototype of veterinarian simulator and prototype of simple tracking equipment. At the end of the second term, we will finish our application and deploy to Play Store.

Chapter 2

Literature Review and Related Theory

2.1 Background

Having pets in the house is more popular in these days even in Thailand, due to the fact that the present state of society has changed. Large family with many members in the same house has been gone and become a smaller family because people want to be more convenient in many ways. Bringing pets to a family is one of many easiest way to relieve their living that more solitary. However, there are many problems that some family doesn't have enough funds and time to take care their pets because the rise of living expenses in both owners and pets including the knowledge they have that can't decide what time to meet a veterinary and sometimes it's too late to cure them.

From our perspective, we realize how important of caring pets is. So we introduced a project called "YellowVet — Smart Veterinary in your pocket". It acts like a virtual veterinarian that can diagnose its illness and report their daily activity. By our product will learn many past animal treatment cases studies to understand what users want to inform (Symptom understanding) and it can be text. These inputs will pass through our machine learning model and generate into the probability of their sickness. There is tracking system that observe an animal behavior for more detail when sending a new answer to our application. Furthermore, YellowVet is a mobile application that is very portable and suitable for modern people. Firstly, we will implement for dogs and will add more pets later in the future.

2.2 Programming languages or libraries, software tools and environments

- **Programming languages**
  - **Java**

    A programming language that is based on object-oriented design. Objective is to let developers write once and run everywhere in different environment as we know as Java virtual machine (JVM). In Android, it uses Java mainly in across any devices.

  - **Kotlin**

    It's mainly designed from Java language and improve some useful advantage from new programming language. Google said that it can fully 100% compatible to Java and reduce 40% of source code, and they recommend to use Kotlin to implement a new Android application [2].

  - **Python**

    It's a high-level programming language that is flexible to run in any platform. Mainly use in data science due to having many machine learning library.

  - **XML**

    It's a markup language that defines a general guideline. In Android, we use these to define a user interface in all activity.

- **Libraries**
  - **Firebase**

    Firebase is a mobile and web application development platform developed by Google. The Firebase platform has 18 products including Realtime database and ML Kit [3].

  - **Keras**

    It's an open-source neural-network library in Python which is running on subset of TensorFlow [4].

- **Tensorflow lite**

  TensorFlow Lite is a minimal TensorFlow library which is more lightweight and focus on mobile and embedded devices. It supports on Android, iOS, and Linux. In theory, it converts the full TensorFlow model into a smaller one and optimize for smaller devices. We do not need to create or train TensorFlow Lite model, we just convert TensorFlow or Keras model to TensorFlow Lite and use it as our predictive model for mobile [5].

- Software tools

- **Android Studio**

  Android Studio is the official integrated development environment for Google's Android operating system, built on JetBrains' IntelliJ IDEA software and designed specifically for Android development [6].

- Hardware

- **RaspberryPi**

  Raspberry Pi is a small single-board computer. It's cheap if we compare with desktop computer. It different from arduino board. Arduino board is microcontroller but raspberry pi is small computers have CPU, RAM, ROM, and OS [7].

- **ArduinoUno**

  Arduino Uno is a microcontroller board based on ATMega328 and develop by Arduino.cc. The board equipped with digital and analog I/O pins that may be connected to other circuits. The board programming with the Arduino IDE (the program developed by Arduino.cc) [8].

- **NodeMCU**

  NodeMCU is IoT platform it includes firmware which ESP8266 system. ESP8266 used to connect Wifi and Firebase by FirebaseArduino

library and easy to connect with another sensor or module in circuit board [9].

- **GY-291**

  GY-291 is a sensor board based on ADXL-345. ADXL-345 is 3-axis accelerometer module with SPI and I2C interface. It measures acceleration of gravity and detect lack of motion compared with 3-axis with user setting [10].

- **HC-06**

  HC-06 is a arduino bluetooth module. It enables the Arduino to be connected and exchange data with other devices such as Smartphone, computer or other microcontrollers. For this project, we use this module to exchange data between arduino IoT and Smartphone application [11].

- **Node32 lite**

  Node32 lite is IoT platform. It includes firmware which ESP8266 system and BT 4.0(Bluetooth low energy). It developed from ESP8266 and include bluetooth and have GPIO 30 ports [12].

- Techniques & Theories

- **Machine Learning**

  Machine Learning is a technique which make computer able to learn by using data. It can separate to 3 main types which are Supervised Learning, Unsupervised Learning and Reinforcement Learning. In this project, we use Supervised Learning technique to create a predictive model. This technique will learn from input data. Moreover, input data for this technique need to has “label” as answer to be objective of this model. Predictive model will learn, compute and adapt to understand the pattern of input data depend on kind of algorithm of model.

- **Natural Language Processing**

The main concept of natural language processing is to make machine learning able to learn or computer input data in string format. It not possible to train machine learning model by input raw string as train data, because it can't recognize characteristic or pattern of train data (sentence which create by combining many words is often unique). The simple solution is to use tokenizer to tokenize sentences in the word, subword or character and preprocessing them.

- **Tokenizer**

  Tokenization is the identification of **linguistically meaningful units (LMU)** from the surface text. For English, it's simple because each LMU is delimited/separated by whitespace. However, in other languages, it might not be the case. For most Romanized languages, such as Indonesian, they have the same whitespace delimiter that can easily identify an LMU. However, sometimes an LMU is a combination of two "words" separated by spaces. E.g. in the Vietnamese sentence, you have to read thời_gian (it means time in English) as one token and not 2 tokens. Separating the two words into 2 tokens yields no LMU or wrong LMU(s). Hence, a proper Vietnamese tokenizer would output thời_gian as one token rather than thời and gian [13]. For Thai language, orthography has no spaces to delimit "words" or "tokens". In this project we will use tokenizer API.

- **Deepcut - True corporation API**

  Deepcut is Thai tokenizer which applied deep learning technique to predict which character is starter of word (binary classification). It will tokenize word since starter of word until starter of next word [14].

- **Maximum matching - PyThaiNLP API**

Maximum matching is an algorithm which check word from dictionary and find the shortest possible segmentation of input sentence [15].

For example

(1) “ไปหามเหสี” → “ไป | หา | มเหสี” word count = 3

(2) “ไปหามเหสี” → “ไป | หาม | เห | สี” word count = 4

Both examples are result of possible segmentation of input sentence. This algorithm will give result as (1) because the word count is shorter than (2).

- **Preprocessing**

One-hot encoding is a technique that represent words that appear in a sentence. Assuming that we have 50 interested words, we will have 50 array indexes and each index refer to each interested word. If interested word appear in a sentence, the space which refer to that interested word will be 1.

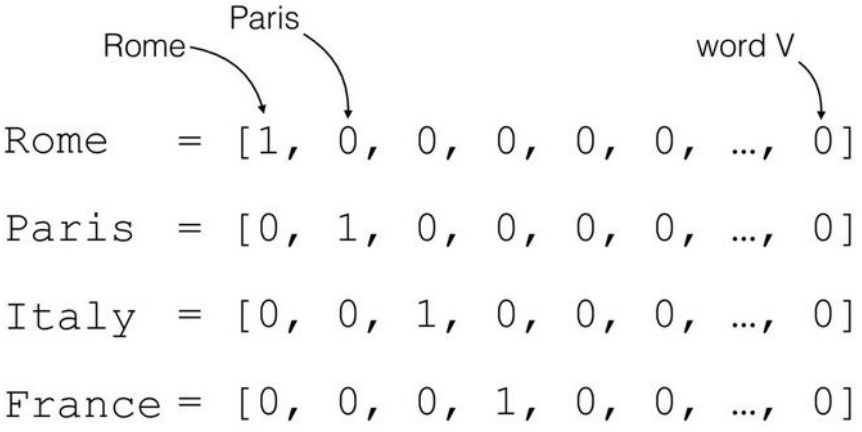


Fig 2.1 : One-hot encoding example [16].

The main benefit of this technique is that we can scope only features or words that we think it is effective and reduce some

unnecessary computing. However, we may be missing some features which are effect to the pattern because we do not think it is important.

Term Frequency Inverse Document Frequency is a text preprocessing method which using numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus [17] . This method is able to find what is “important word” which can refer to each symptom and it will ignore “common word” which does not give any information.

This technique will weight each word, we can understand how important each word. The weight will get by learning word in dataset.

Word embedding is a text preprocessing method which is the collective name for a set of language modeling and feature learning techniques in natural language processing (NLP) where words or phrases from the vocabulary are mapped to vectors of real numbers by considering similarity of nearby word frequently occur in the train dataset [18].

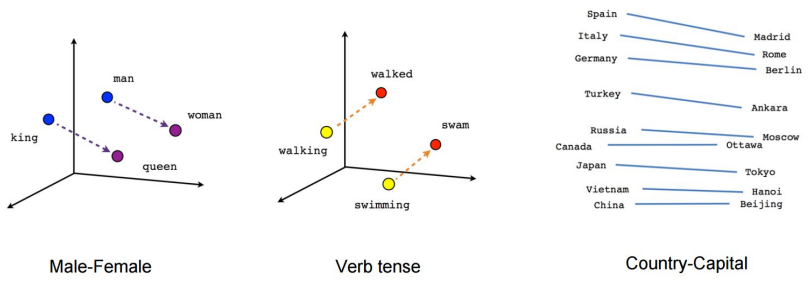


Fig 2.2 : Word embedding example [18].

This technique can tell similarity between word by location of vector. It is very useful if we have a large dataset. We can also use pre-trained embedding weight which is embedding vector file that store location of each word that learn from other dataset.

- **Naive Bayes**

Naive Bayes is a simple, effective and commonly-used, machine learning classifiers. It is a probabilistic classifier that makes classifications using the Maximum A Posteriori decision rule in a Bayesian setting. It can also be represented using a very simple Bayesian

network. Naive Bayes classifiers have been especially popular for text classification, and are a traditional solution for problems such as spam detection [19].

- **Multilayer Perceptron**

  Multilayer Perceptron is an artificial neural network which more than one layer. It has “ Input layer ” for receive data signal and “ Output layer ” for predict result from input layer. Between these two layers will has “ Hidden layer ” which able to compute in multilayer perceptron.

  Multilayer perceptron is very popular for supervised learning by training pair of input-output and learn correlation between them. It can tune parameters such as weights and bias of model to minimize the error.

  Backpropagation is always used for adjust weight and bias by observing error. Error can be observed by using many metrics depend on task and objective such as Root Mean Square Error. Backpropagation is an expression for the partial derivative $\partial C/\partial w$ of the cost function C with respect to any weight w (or bias b) in the network. The expression tells us how quickly the cost changes when we change the weights and biases [20].

- **Long Short Term Memory (LSTM)**

  LSTM is classified as Recurrent Neural Network which is an artificial neural network which using output of previous state to compute in current state. It is a Recurrent Neural Network that designed for solving Vanishing Gradient Problem. Vanishing Gradient is problem that gradient (∂) is fewer or vanish after get back propagation many times. LSTM is solved this problem by learning weight of each node and has Gate cell which change every time at each node [21].

2.3 Related Research

2.3.1 Your.MD

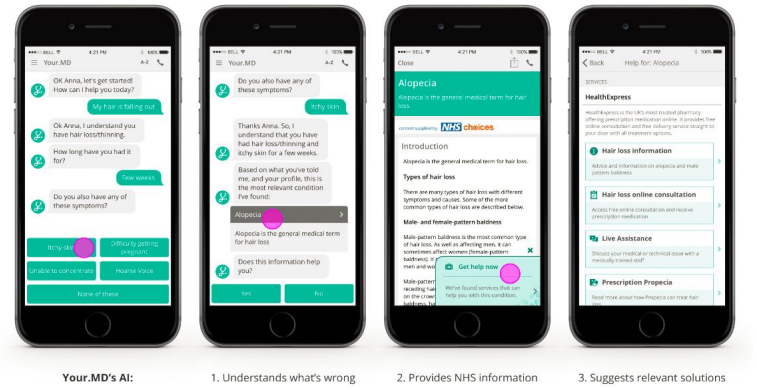


Figure 2.3 Analyze Symptom by Using Your.MD

By using AI and machine learning, Your.MD is a doctor chatbot which can help you find your health information. Algorithms trained on "validated medical literature covering over 1000 medical conditions" allow the chatbot to learn common illness and provide recommendations for relevant resources. At the moment their illness Checker can calculate over 250 different conditions. User will fill input as text and Your.MD will make a conversation [22].

Comparing with our project, this project is chatbot but our project is similar to text answering system which easier to keep history of inquiry and allow user to tell anything what they observe in their pet, but chatbot still more popular because it can add many techniques to make conversation with user and it has specific question which easier to make an illness classification, but the disadvantage of making customize chatbot is it consumes a lot of time or it needs well design of conversation to specific scope of illness. The last difference is our project is designed to use Thai language text as input. Moreover, we have behavioral tracking device to get more information to diagnose illness.

2.3.2 Sensely

Figure 2.4 Using Sensely to access health and care advice

Sensely is Virtual Medical Assistant avatar integrates AI to recommend diagnosis based on patient illness using a smartphone. The platform uses algorithms trained on large volumes of clinical content, such as medical protocols and chronic disease information, to interpret patient illness and to recommend an appropriate diagnosis. Patients can describe their illness to the virtual medical assistant named Molly, using speech, text, images and video [23].

Sensly is very similar to Your.MD but it adds more functions and can receive more type of input,i.e. video, speech or image.

Chapter 3

Design and Methodology

3.1 Veterinarian Simulator Experimental Design

3.1.1 Data

Data of this project came from pet community webboards which allow pet owners to create topics or posts about their pet problem. Our selected webboards must have reliable answer which came from real veterinarian.

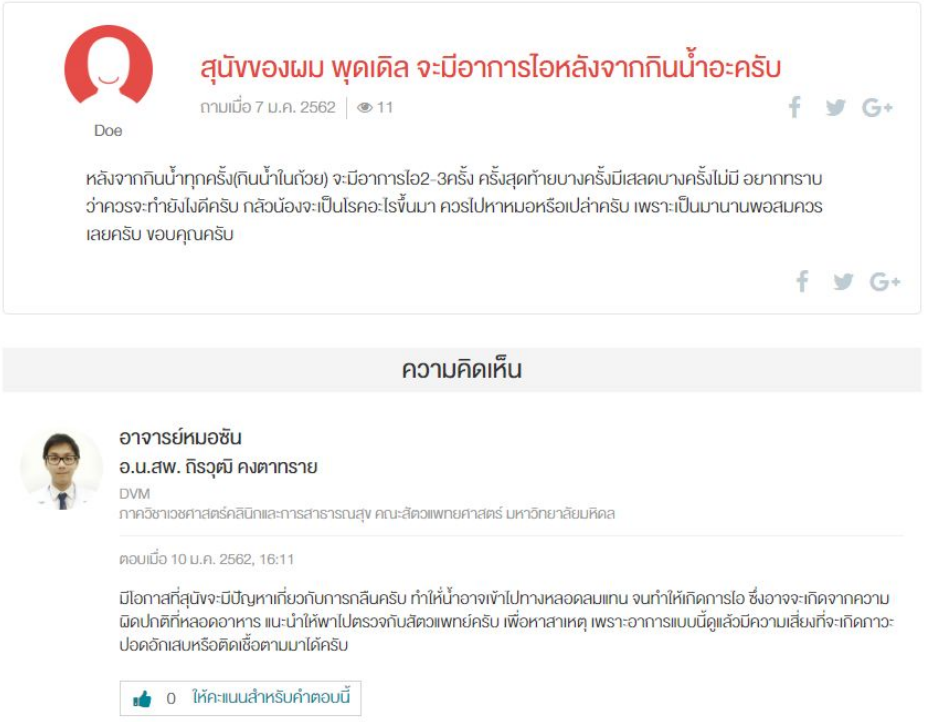


Figure 3.1 Example of post from pet community webboard [24]

Our data will be in comma separated values format which contain 2 columns. First column is “ Question ”, it represents question which pet owner ask in pet community webboard. Second column is “ Answer ”, it represents a type of illness which came from summary of veterinarian’s answer to this question. This column we summarized by ourselves. In current progress, we have 12 illness types.

Question	Answer
ไอแห้งๆมีเสียงดังเหมือนพยายามจะเอาอะไรออกมาจากคอ มีฟองสีขาวเหนียวๆ	พยาธิ
อาการเป็นผื่นแดงๆขึ้นตามตัว และมีเป็นตุ่มน้ำขึ้น ครับ	โรคผิวหนัง
อาการคือกินข้าวไม่ได้มาหลายวัน จนวันนี้ นอนนิ่ง อ่อนแรง คอแข็ง ตาลอยพยายามให้กินนมเพื่อล้างพิษ พอกินแล้วก็รู้สึกตัวขึ้นมา นิดห	ติดเชื้อ
อาการของสุนัขพันธ์ปอมหน่อยค่ะ มีอาการนอนแล้วโก่งก้นขึ้นบ่อยๆ เป็นอะไรหรือป่าวคะ เขาจะซึม หนูเครียดมากเรยค่ะ	ปัญหาพฤติกรรมสัตว์
อ้วก แล้วก็ถ่ายเป็นเลือดคะหมอ	ระบบทางเดินอาหารผิดปกติ
มีก้อนมันขึ้นตรงนมน้องค่ะ เหมือนเป็นสะเก็ดแผล นูนๆด้วยค่ะ	ก้อนเนื้อผิดปกติ

Figure 3.2 Example of data in Question-Answer format

The “Answer” column contains the name of 12 illnesses that our veterinarian simulator is able to diagnose which are

1. ก้อนเนื้อผิดปกติ (Abnormal lump)
2. ความผิดปกติของทางเดินหายใจ (Respiratory disorders)
3. ติดเชื้อ (Infected)
4. ท้องผูก (Constipation)
5. ท้องเสีย (Diarrhea)
6. ปัญหาพฤติกรรมสัตว์ (Animal behavior problems)
7. พยาธิ (Parasite)
8. ระบบทางเดินอาหารผิดปกติ (Gastrointestinal disorders)
9. โรคทางข้อกระดูก (Bone disease)
10. โรคทางระบบประสาท (Neurological diseases)
11. โรคทางสายตา (Eye disease)
12. โรคผิวหนัง (Skin diseases)

3.1.2 Method

We use natural language processing (NLP) technique to make our veterinarian simulator understand the input text by tokenizing and many text preprocessing techniques. Our veterinarian will learn pattern of treatment cases in past by using machine learning models such as deep learning, MLP or likelihood model.

3.1.2.1 Dictionary & MLP Classification

The first experiment start from tokenize “Question” in form “list of word”. After that, we will create a dictionary of word from all questions in dataset. Dictionary of word will store “Key” and “Word”. Key in dictionary is incremental value that represent each word and word in the dictionary is come from every unique word in “Question” column in train dataset.

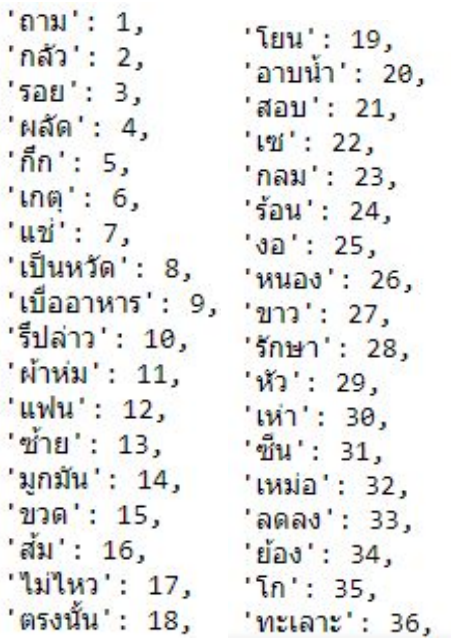


Figure 3.3 Example of dictionary of word

Next step is to replace words in “ list of word ” with “ Key ” value in dictionary of words.

For example : [‘อาบน้ำ’, ‘เป็นหวัด’] → [20, 8]

And we need to make every “ list of word ” to be equal in length by define max length and fill with zero until complete, in this experiment we define max length as 50 which came from average length of question in train data. This method is very popular for text classification task.

For example : [20, 8] → [0, 0,0, … , 20, 8] In the last step, we will use Keras library to create multilayer perceptron model which train from pair of “Question” and “Answer” in train data. “Question” will be in form array of numbers that represent each word. For multilayer perceptron, we will put Embedding Layer that able to create embedding vector which can represent

meaning (similarity) between sentences and also has LSTM layer which learn order of the word and extract the meanings.

3.1.2.2 One-hot Encoding & MLP Classification

The second experiment is different from the first one in that process before train in multilayer perceptron. Instead of convert text to dictionary, we decide to define our interested symptoms, and we set rule-based when detect word which can represent each symptom. Our “ Question ” will be converted in term of one hot encoding array which each position of binary represents each symptom.

For example : [‘ร้อง’, ‘ส่งเสียง’, ‘หอน’] these words will represent “Cry” when found these words in “Question” array position of “Cry” will be 1. For current progress, we have 24 interested symptoms.

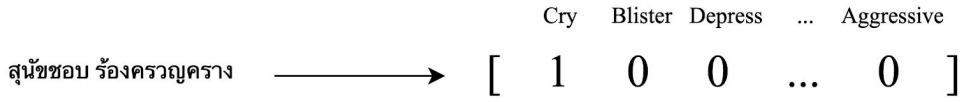


Figure 3.4 Example of binary array of symptom

Our 24 interested symptoms which we use as input of illness classification models are

['Cry', 'Blister', 'Depress', 'Lesion', 'Flea', 'Pus',

'Not_walk', 'Diarrhea', 'Constipation', 'Blood_waste', 'Rash', 'Vomit',

'Fat', 'Skinny', 'Pregnant', 'Breathing', 'Not_eat', 'Aggressive',

'Weak', 'High_temp', 'Vibrate', 'Bone', 'Eye', 'Not_react'].

In the last step, we will create multilayer perceptron model which train from pair of “ Question ” and “ Answer ” in train data. “ Question ” will be in form binary array of 24 symptoms.

3.1.2.3 Symptom Understanding & MLP Classification

The third experiment is very similar to the second experiment, but instead specific rule-based for each symptom we create another classification to predict symptoms from “ Question ”. Using this method we need to label symptoms to use as label of train data. Next step is to create predictive models which can predict symptoms from “ Question ”, we called this step as “ Symptom Understanding ”. For predictive model in this step, we use multilayer perceptron which has both Embedding Layer and LSTM

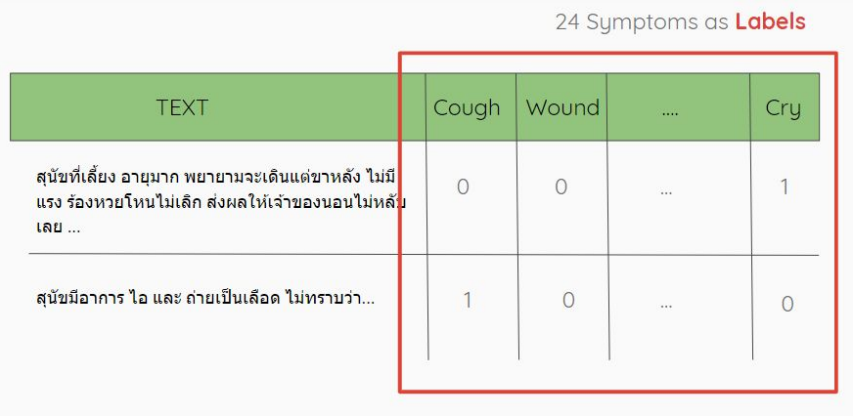


Figure 3.5 Example of train data for Symptom Understanding

After this step, we will get output in form of probability in each symptom. Next step, we will set acceptable threshold to change the probability to binary format. If probability of symptom more than threshold, the value of that symptom will be 1. In the last step, we will create multilayer perceptron model which train from pair of “ Question ” and “ Answer ” in train data. “ Question ” will be in form binary array of symptoms. It means we have two predictive models, one for symptom and another one for illness.

3.1.2.4 TF-IDF & Traditional model

The last experiment use different preprocessing technique. Term Frequency Inverse Document Frequency is preprocessing technique, it will weight or give different value to each word. For this experiment, we will use traditional model like “ Naive Bayes ” to classify illness. In the last step, we

train naive bayes classifier model by pair of “ Question ” and “ Answer ” in train data.

3.1.3 Evaluation

We evaluate the performance of veterinarian simulator by using validation set which accepted by our veterinarian consultant. Validation set will contain data in Question-Answer format as same as train data. Metric that we chose for experiment is *diagnosis score*. Our veterinarian simulator will generate output as probability of each illness, and we plan to show 3 illnesses which get the highest probability as result. In validation performance step, if the true answer illness (validation by real veterinarian) is in top 3 illnesses which our veterinarian simulator predict, our veterinarian will get *diagnosis score*. It’s weak criteria for evaluating the machine learning model. However, our objective is to encourage users to read the most 3 illnesses which user’s pet may have. Then, we decide to use this metric to evaluate our model performance.

3.2 Software

3.2.1 Use Case diagram

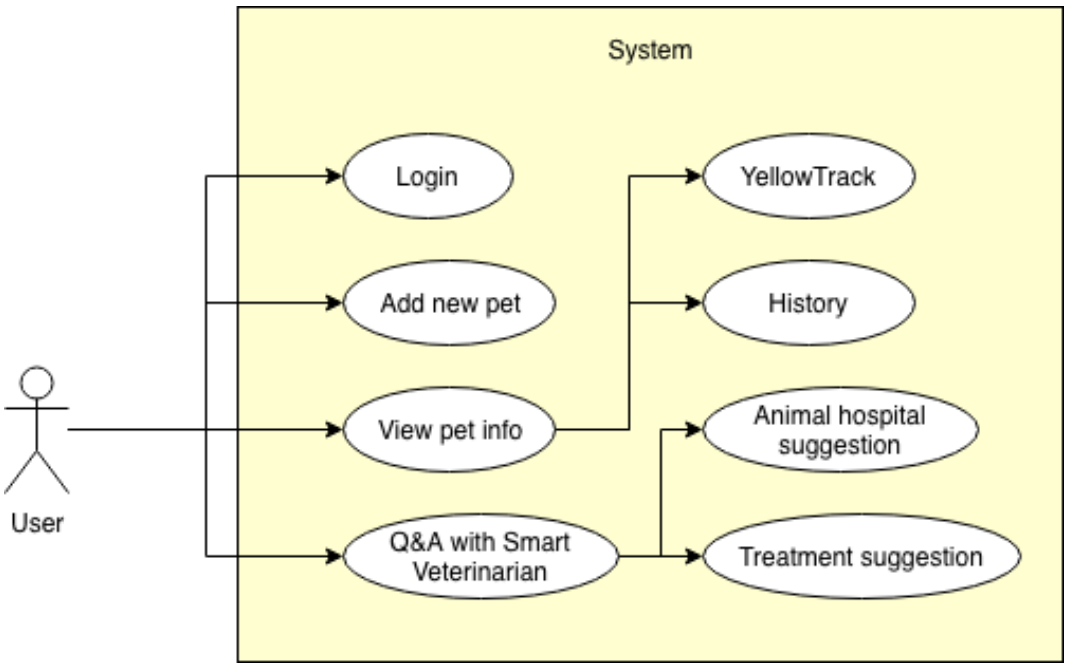


Figure 3.6 Use Case diagram

YellowVet is a mobile application that provides many functionalities for users. They can login to their account which shows all of your pets. When user has a new pet, they also can add it to their account. View their pet information including YellowTrack which is tracking all of pet's activity and also history as well. Last important function is asked question to system and diagnosis the symptom they ask. After they put text to the system, it will suggest nearby hospital and some basic treatment to user.

3.2.2 User interface screen layouts

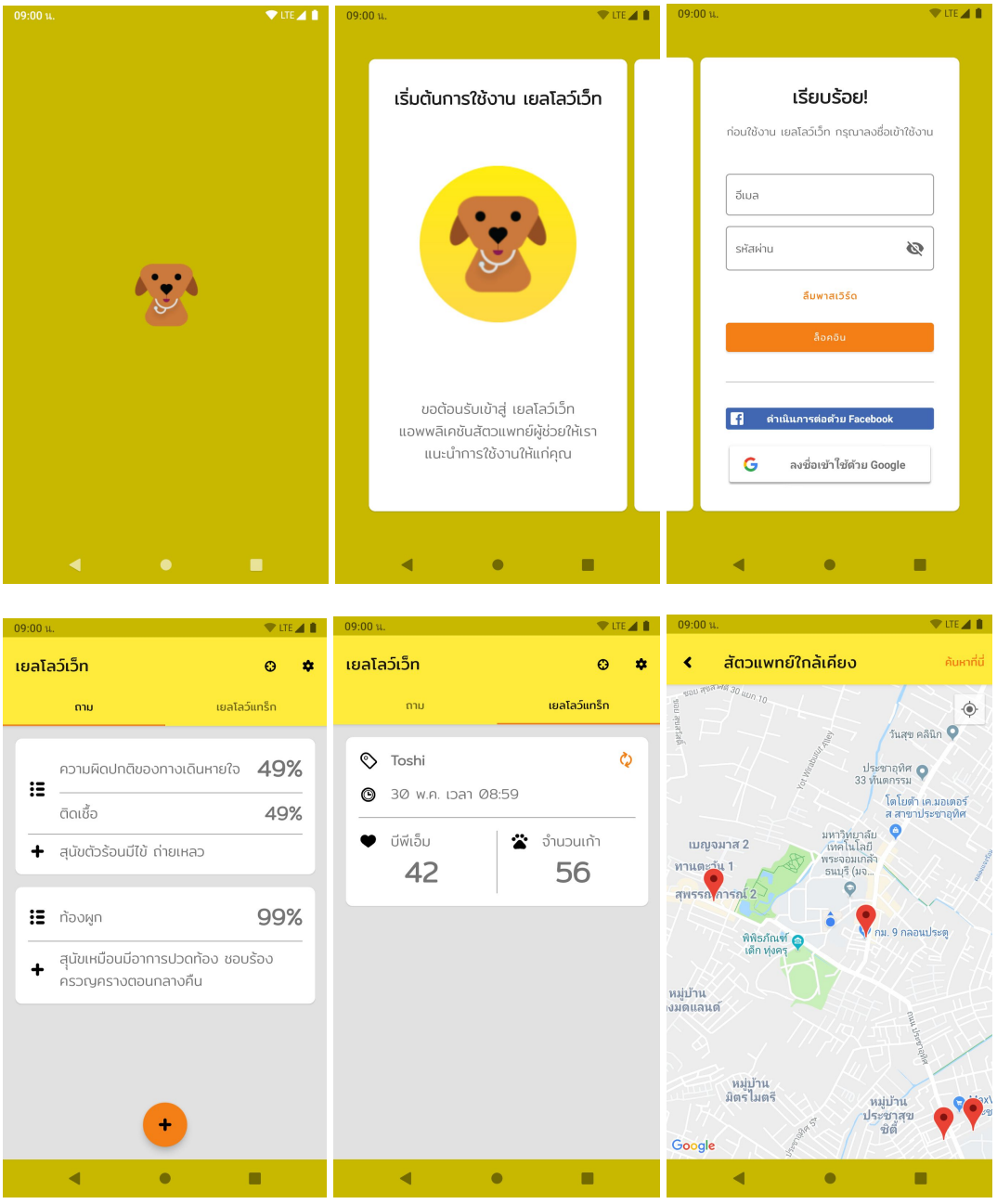


Figure 3.7 YellowVet user interfaces

From our perspective, user experience is also important in this project. We want to make our application in the best quality. Nowadays, there are many design theory in the world and one we choose is “Material design” which officially made by Google. We use this library because it’s widespread and looks seamlessly to almost Android application.

Material design is design language for user interfaces. There are mainly available in Android, iOS, and web application. The foundation of material design is using the basic shape such as rectangle, square, circle combined into a new shape. We use these theories in designing the icon too.

Figure 3.8 YellowVet application icon

After users open our application “YellowVet”, they will see the upper tab which is a name of current pet you use. Users can switch to another pet or add a new pet by pressing a button right next to a pet’s name profile.

Down next to the pet’s name profile, there are two major tabs including:

1. Ask — this tab will gather all answers that users have been done before and you can add new symptom with text to make a result more accuracy. After you add new symptom, application will show you a result including the probability of sickness, general information of this disease, how owners treat them, etc.
2. YellowTrack — this tab will be available with our hardware system which observe user’s animal behavior. For example, if pets are dogs, our hardware will collect steps, track a location, etc.

3.2.3 How we plan the communication for “Ask” scenario

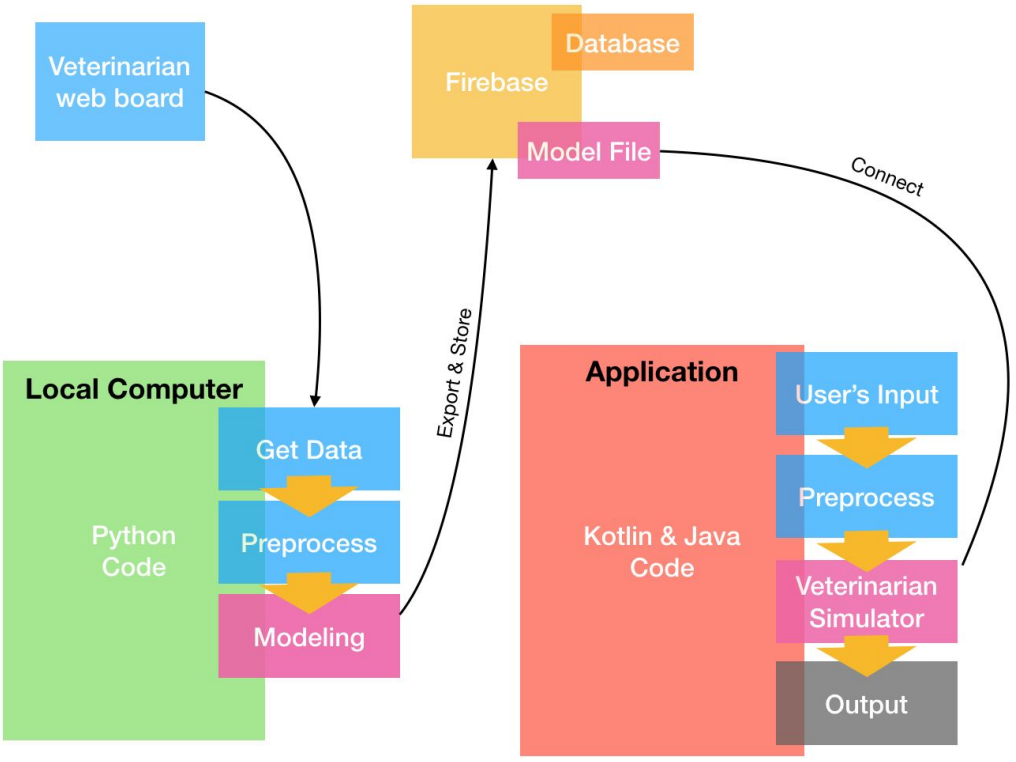


Figure 3.9 Overview of communication in this project for “ASK” scenario

After creating model in our local computer, we will export it and put it on Firebase. When user want to ASK, our application will use similar preprocessing technique which we do when we created the model. The preprocessed text will use as input of veterinarian simulator which is our stored model on Firebase that hosts a model on cloud and acts as API layer for an application’s usage. YellowVet always keep sync between model on both cloud and local device up-to-date. Everytime users open our application, it always use an updated model on their mobile device and when they input some text, our model on local device will return the probabilities of each illness as output to our application.

3.3 Hardware

3.3.1 Overview

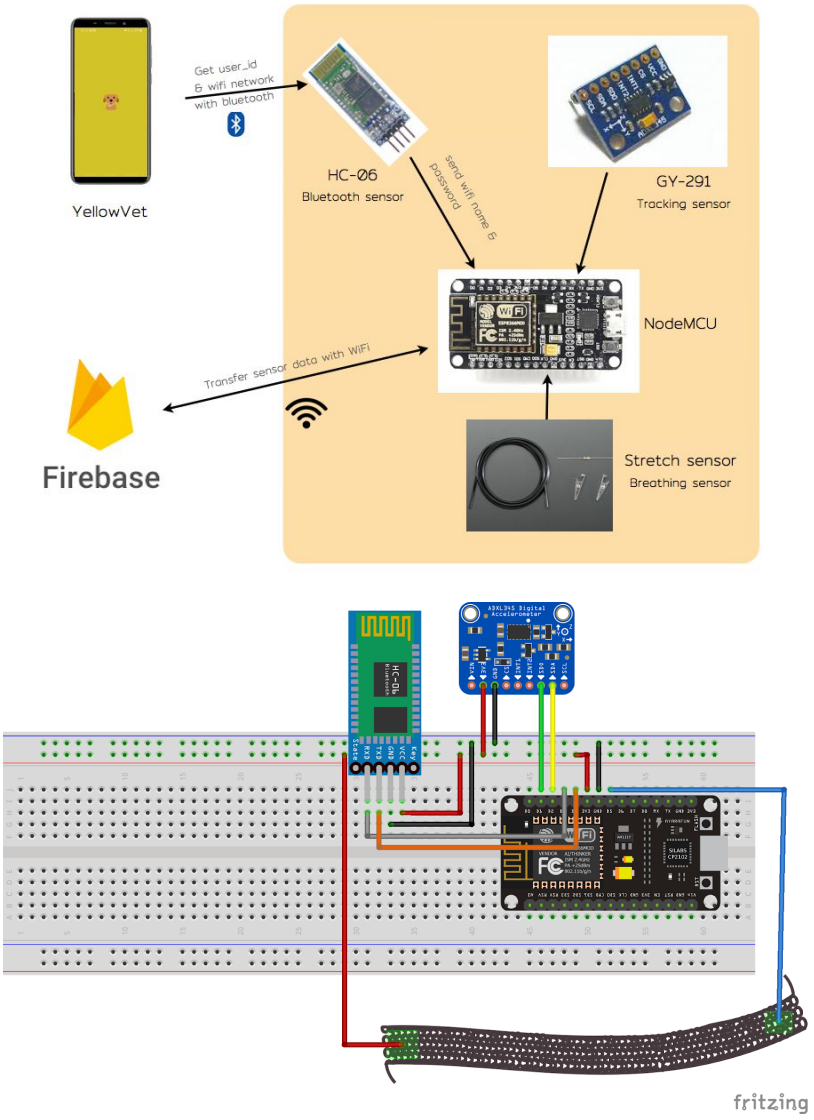


Figure 3.10 Overview of Hardware Gadget

In Hardware part (Figure 3.9), our gadgets connect to application with bluetooth(HC-06) to send userID and use WiFi network to connect Firebase. When we get WiFi name and password from application, then nodeMCU will connect Firebase to send and receive data from Realtime database. In part of sensor, we divide into 2 parts which are accelerometer sensor and breathing sensor. Accelerometer sensor used to measure movement and proper acceleration. Breathing sensor which created by resistor knitted with fiber to measure flow of electricity. Both sensors will send data to Arduino UNO to collect data and send to Firebase. In

part of accelerometer, we use GY-291 to measure step counting by detect lack of motion compared with 3-axis with user settings.

3.3.2 Flow chart

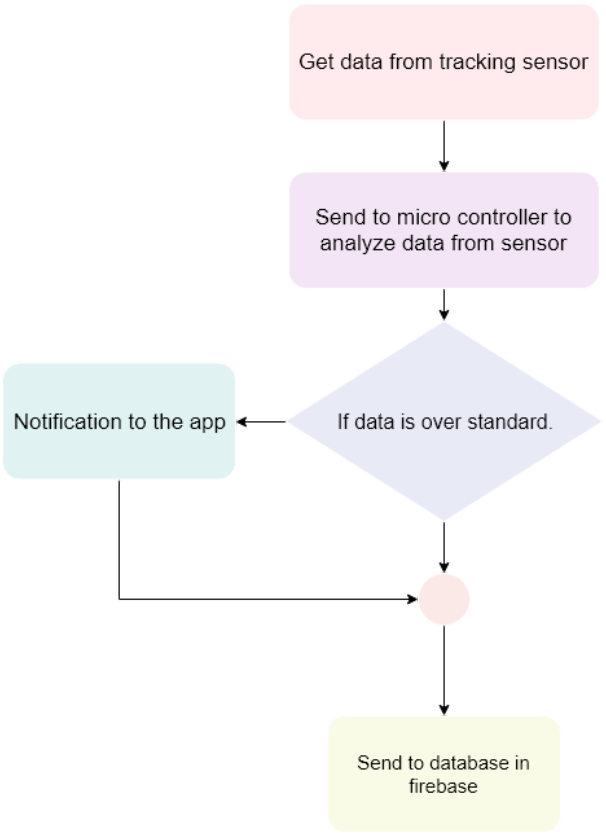


Figure 3.11 Flow chart of Hardware Gadget

YellowTrack is IoT gadget in YellowVet application. First, It gets data of 3-axis from GY-291 accelerometer and breathing counter in 1 minute and send it to microcontroller to analyze data from sensor. When microcontroller check data if data is over standard, microcontroller will send data to application and notice on it. If data is on standard, It will send data to realtime database in Firebase.

3.4 Firebase Database

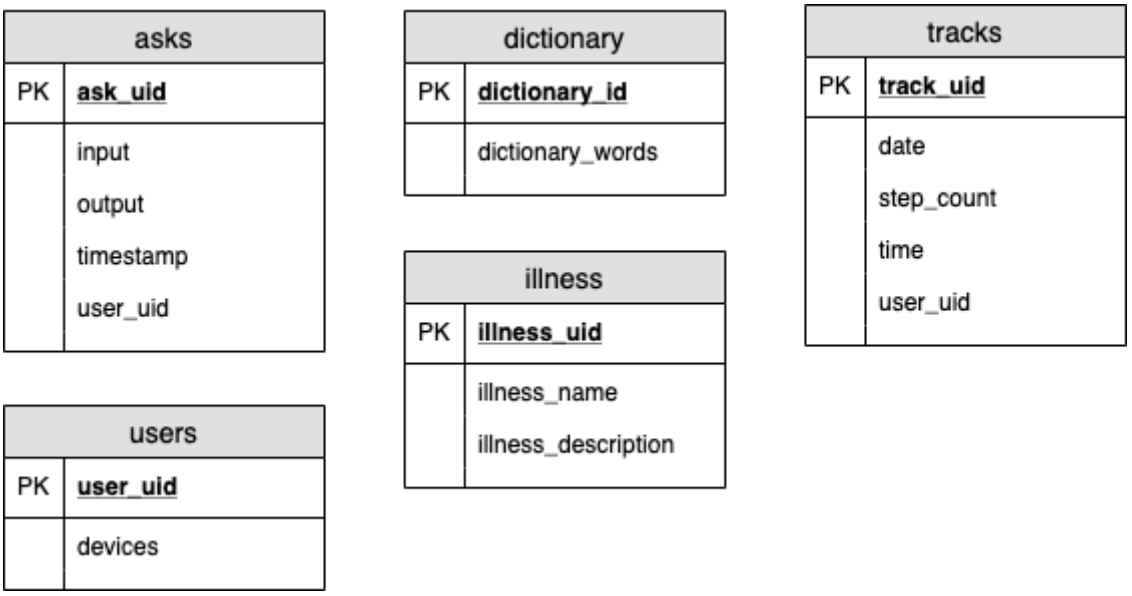


Figure 3.12 ER diagram

- **asks** = Keep all questions that users have asked.
- **illness_table** = All illness information to retrieve in application.
- **tracks** = All tracking information including step_count and user_uid.
- **users** = All users information such as YellowTrack devices.

Chapter 4

Results and Discussion

4.1 Veterinarian Simulator

Table 4.1 *Diagnosis score* ** for veterinarian simulator experiments

Experiment ** Higher *diagnosis score* is better	Dictionary & MLP	One-hot & MLP * Not use Tokenizer	SympUn & MLP	TF-IDF & Naive Bayes
train data : 200 records Tokenizer : deepcut	0.2820	0.3589	0.2820	0.3589
train data : 200 records Tokenizer : maximum matching	0.2820		0.2820	0.2820
train data : 1000 records Tokenizer : deepcut	0.6923	**0.7948**	0.4358	0.3589
train data : 1000 records Tokenizer : maximum matching	0.5897		0.5641	0.3333

We do this 4 experiments including some interesting factors which are amount of training data, tokenizing algorithm. The result was going on as expected, If we increase the number of training data, the accuracy will increase as well. For experiments that we use only 200 records as training data, many experiments get diagnosis score at 0.2820 which means our veterinarian just predict the most occurred illness ("*Abnormal lump"* for this case) in training set for every case to obtain the highest score. It is a common problem for supervised learning in unbalanced dataset.

After we increase the number of training data to 1,000 records, In part of Dictionary & MLP experiment shows the differences between tokenizer. We can see that diagnosis score of deepcut tokenizer is significantly higher in "Dictionary&MLP" and "TF-IDF Naive Bayes" experiment because the sentence that was tokenize by deepcut will have shorter word compare with maximum matching. LSTM or RNN type layer will learn from pattern of order of word. Then, if we compare between short word and long word, we can see that shorter word order will have more chance to be "Pattern". The model will use benefit from this pattern to get more information to classify. However, in Symptom Understanding the maximum matching tokenizer is perform better because in this experiment we don't have enough data for each symptom. Then if we use maximum matching that cut the sentence to longer word than "deepcut", it will be easier to detect the pattern of training data.

For the second experiment or One-hot & MLP, the result is significantly high. This method helps model select the important features which is good technique if we do not have enough of training data. However, this technique may give disadvantage if we miss some important features, our veterinarian simulator will lack features that can diagnose some illness that relate to those features. Another disadvantage is that this method uses our language knowledge more than other methods. If we have more data, this method may stick with our knowledge boundary and can't perform well when compared with other methods.

In SympUn & MLP experiment, the result seem not good enough because the Symptom Understanding part is required large data to train this part. If we want to use this technique we may need more than 10,000 records because it is multi label classification and it has too many labels to classify.

For the last experiment which is TF-IDF & Naive Bayes, seem like the result is not as good as expected. By using statistical models like naive bayes it not helps us do the feature engineering like MLP or other neural network models. Then, using only Tf-Idf may not get enough information to make the decision.

In conclusion, we use the second method or one-hot & MLP for current version because this method is very effective for small dataset. After we let some user play our application, from our opinion we think the result is acceptable. But it still has some cases that our veterinarian simulator gives the wrong answer. However, if we increase training data, our application will reach higher accuracy.

4.2 Hardware

4.2.1 Tool experiments

- **Raspberry Pi 3**

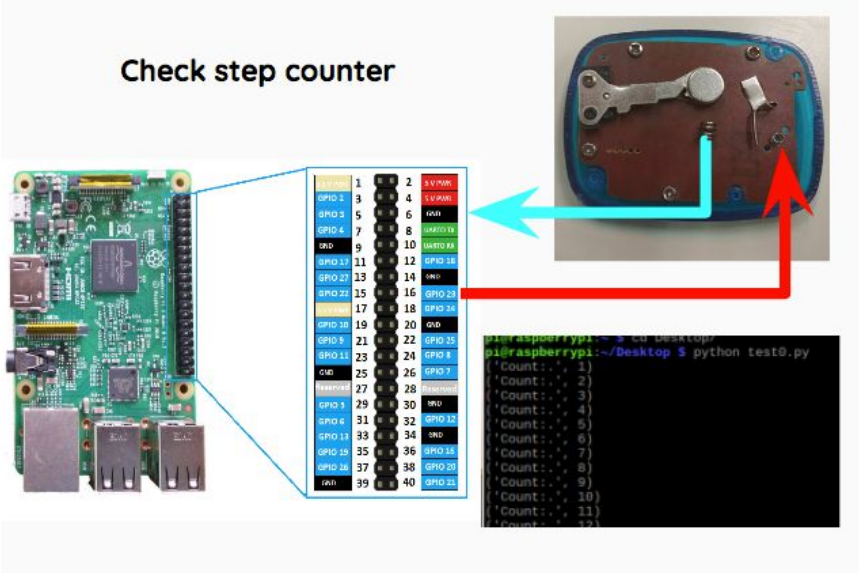


Figure 4.1 Raspberry Pi 3

Raspberry Pi 3 is a small board computer. it can be used to assemble circuit boards with arduino module and connect Wifi. We sample step counting with analog pedometer. it can calculate step counting but it's sensitive to vibration, so it's difficult to use in real usecase.

- **Arduino UNO**

Arduino Uno is a microcontroller board which mainly controlled by ATmega328P. It easy to use and have many useful libraries. However, we didn't choose this tool because nodeMCU already have esp8266 in board, then we change from arduino Uno to nodeMCU.

- **NodeMCU**

NodeMCU has enough library to make this project. The first one is FirebaseArduino.h which is library for communication with Firebase. The second one is AdafruitADXL345.h which can measure 3-axis (x,y,z) from the position of board, we mainly used this library to make step counter. Moreover, it has many comfortable libraries such as SoftwareSerial which can read input wires. Another advantage over 2 tools above is weight. It is suitable to integrate with wearable device.

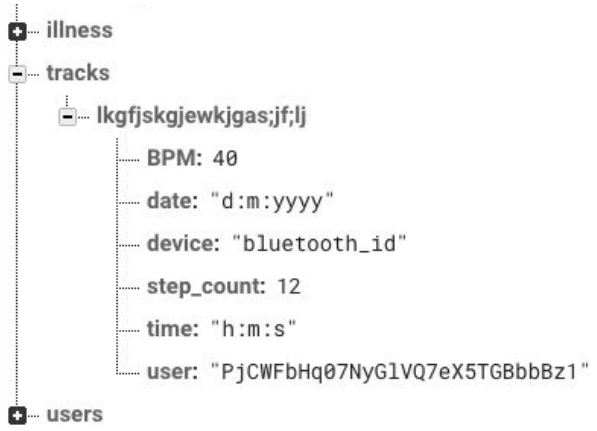


Figure 4.2 All hardware information in Firebase

- **Bluetooth test**

We use HC-06 to connect smartphone with bluetooth. HC-06 is arduino bluetooth module (Figure 4.2). It use to connect between bluetooth and smartphone, computer or other board. The connecting test results show in Figure 4.3 and Figure 4.4.

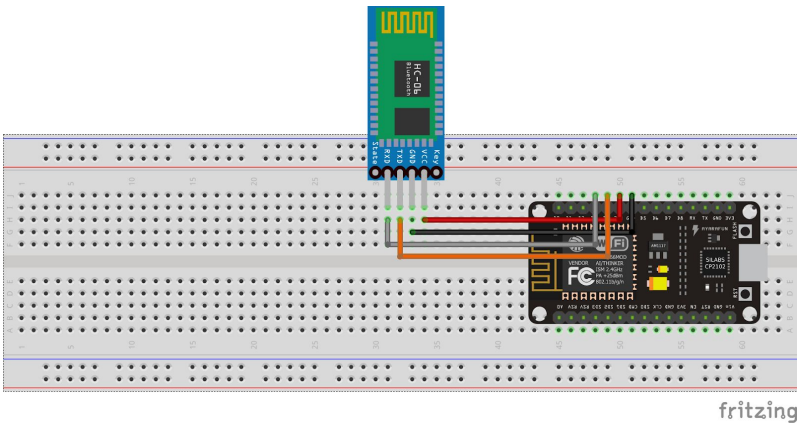


Figure 4.3 HC-06 Bluetooth circuit

Figure 4.4 Result from smartphone when connect and communicate with device

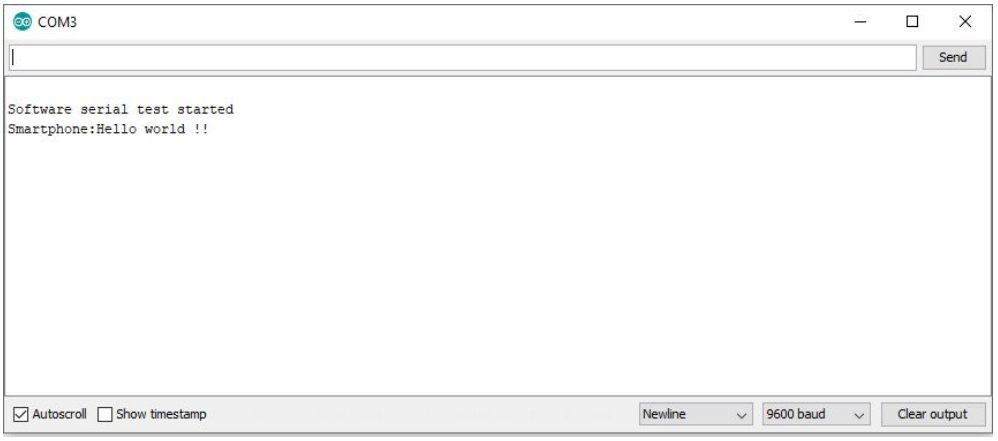


Figure 4.5 Result from computer when connect and communicate with smartphone

- **Step counting sensor**

In this part, we use GY-291 to receive 3-axis and calculate in function. Output of function is step count and send to Firebase with date, time, and user_id. In example (Figure 4.5), we get 3-axis and find the difference between 3-axis at t (current) and t-1 (previous). the step count will calculate from equation (4.1) below.It calculate by getting different accelerate between current time(t) and previous time(t-1). If result get highly different from the standard value, It will calculate by result of different accelerate per standard value for one step (we use this equation in all 3 axis).

$$\text{Stepcount} = \frac{\text{current axis - previous axis}}{\text{distance for one step}} \qquad (4.1)$$

**** In case of distance for one step = 10**

If **a(x) at t** = 10 and **a(x) at t-1** = 0. It means 1 step occurs
Use the same condition with a(y) and a(z)

Figure 4.6 How we calculate step count

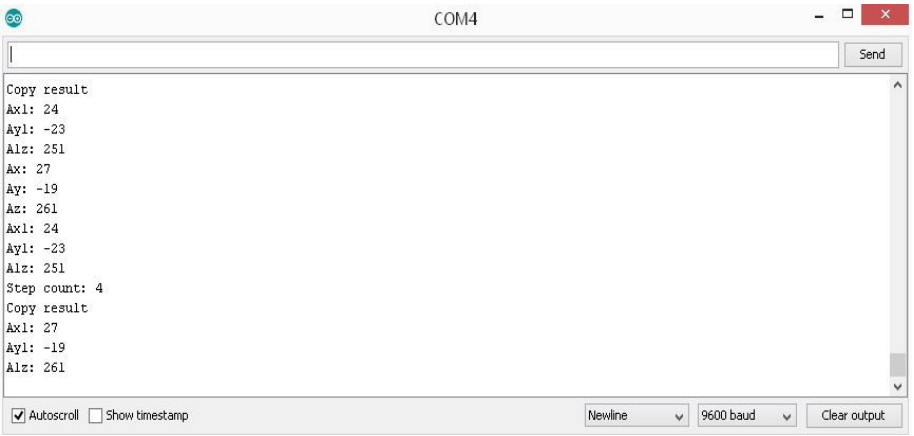


Figure 4.7 Example of result from GY-291

- **Breathing sensor**

In this part, we use stretch sensor to measure the stretchiness (resistance level). Yarn have resistance then we use yarn instead potentiometer(Figure 4.6). We simply test by using LDR (Light Dependent Resistor) to read resistor from board.

Figure 4.8 Measure resistance from yarn

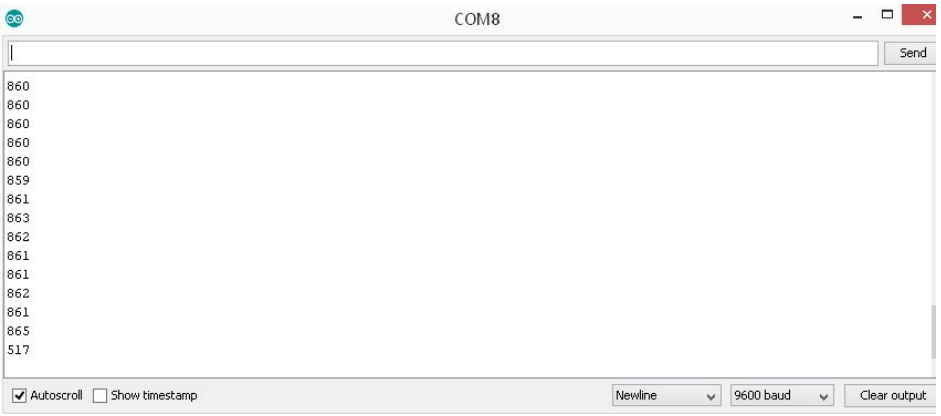


Figure 4.9 Simple test measure resistance from LDR

From picture above (Figure 4.7), 860 ~ 870 is default value of LDR. 517 is value when we decrease light reflect to LDR. We apply this with our breathing sensor. We used yarn as LDR, when yarn stretch the value of resistor will decrease. We will count breathing when value of resistor less than threshold.

4.2.2 Supporting diagnosis

We use result from YellowTrack to help our veterinarian simulator diagnose the illness by set threshold for one-hot symptom. According to

the selected method for this project which is one-hot MLP, we have 24 one-hot encoding symptoms. All symptoms which we have are

['Cry', 'Blister', 'Depress', 'Lesion', 'Flea', 'Pus',
'Not_walk', 'Diarrhea', 'Constipation', 'Blood_waste', 'Rash', 'Vomit',
'Fat', 'Skinny', 'Pregnant', 'Breathing', 'Not_eat', 'Aggressive',
'Weak', 'High_temp', 'Vibrate', 'Bone', 'Eye', 'Not_react'].

We will set rule-based to change symptom to 1 when record from the YellowTrack is more than our threshold or it is in our condition. It will help when user forget to fill some detail about those symptoms. In current progress, we have step tracking and breathing sensor. It means we have 2 symptoms that can affect by YellowTrack which are 'Not_walk' and 'Breathing'.

For condition of ‘Not_walk’ , we set this symptom to 1 when

user’s dog step count today < Avg. step count last week + SD. step count last week

For condition of ‘Breathing’ (or Snuffle), we set this symptom to 1 when

user’s dog BPM of this day > Avg. BPM last week + SD. BPM last week

** BPM stands for Breathing Per Minute

We use Avg. + SD. (standard deviation) to tell that it has significantly change happen (abnormal behavior) as threshold.

Table 4.2 *Diagnosis score* between using and not using 2 YellowTrack symptoms

Experiment	One-hot & MLP 24 symptom	One-hot & MLP 22 symptoms (without ‘Breathing’ and ‘Not_walk’)
train data : 1,000 records	0.7948	0.7435

According to Table 4.2, we do a comparing experiment between use all symptoms and 22 symptoms (except ‘Breathing’ and ‘Not_walk’) without changing other process in previous one-hot MLP experiment. We can see that these two columns have impact to diagnosis score. If user forget to fill information about these 2 symptoms when use “ASK” feature, our veterinarian simulator may misdiagnose some illness. This experiment can show that YellowTrack will make our veterinarian simulator more accurate in diagnosis.

4.3 YellowVet Application

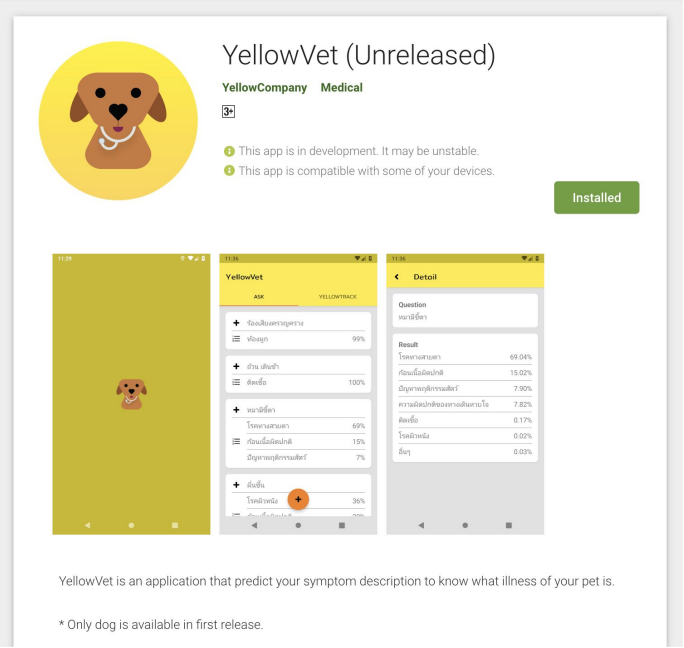


Figure 4.10 YellowVet on Play store

- **Welcome pages**

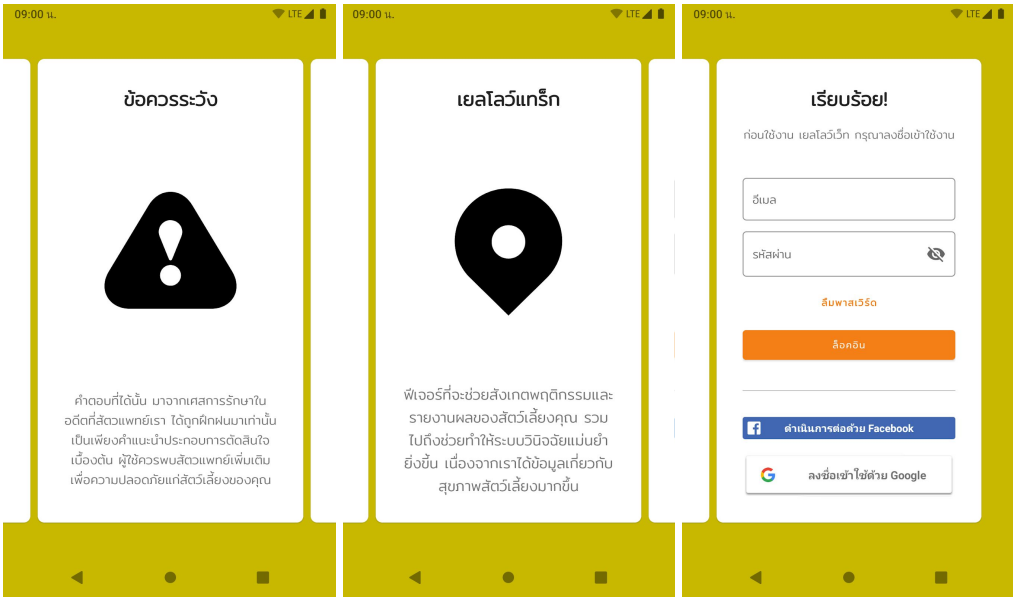


Figure 4.11 Initial launch with welcome pages on YellowVet

When user install YellowVet application for the first time, there are welcome pages to introduce how it works and login page which has many ways to do such as email, Facebook, and Google. Behind the system, we use one of functions in Firebase called “Firebase Authentication”.

- **Firebase Authentication**

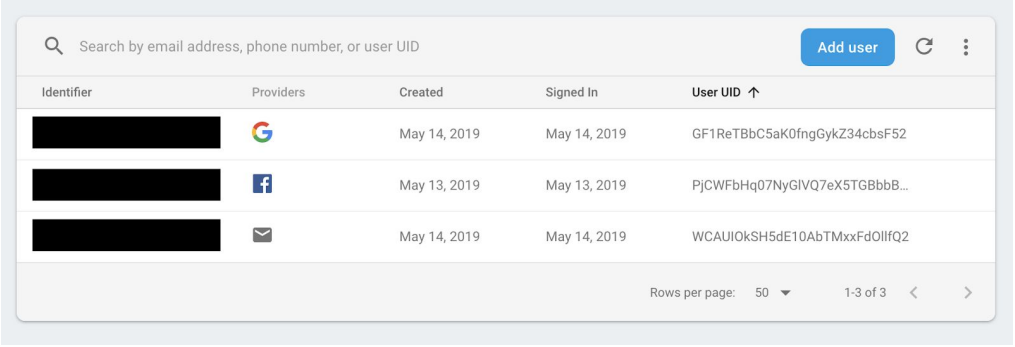


Figure 4.12 Example of Firebase Authentication

Firebase Authentication provides us easy-to-use SDKs and library to make account system. There are many services such as email, phone, Facebook, Google, Twitter, Microsoft, etc. For this project, we use only email, Facebook, and Google that we think it’s enough for general users. When user sign in to the application, it will create user unique id. Then, we use this for database to identify which one is information for each user.

All users data when using Firebase Authentication will keep safety due to the EU General Data Protection Regulation (GDPR). All Firebase customers act as the “data controller” and Google acts as “data processor” that means all customers have the right to respond to their personal data.

In terms of data processor, Firebase Authentication uses passwords, email addresses, phone numbers, user agents, and IP addresses to enable end-user account management such as register, sign in, sign out, and etc. Moreover, Firebase Authentication will encrypt all data through HTTPS and make Google as data processor can’t access users' personal data [25].

- **Connect to model**
  - **Keras and Tensorflow**

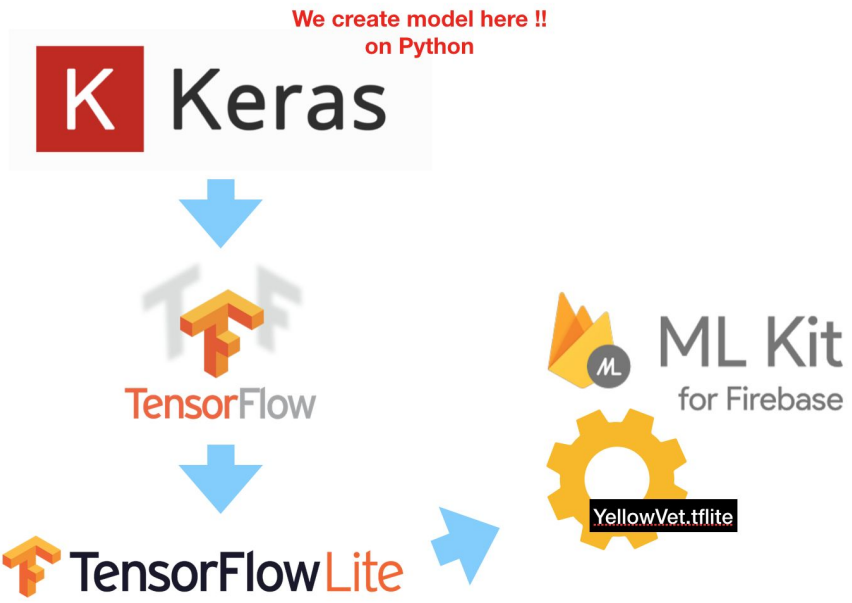


Figure 4.13 How we use model which create on Python work

First of all, we train our veterinarian model on local computer. We use the most popular library for deep learning in python which is Keras to create our model. However, we can’t use Keras model on our application directly. Our solution is to convert Keras model to Tensorflow model. We need to export Tensorflow model to Tensorflow Lite file. Next step is putting our Tensorflow lite model on our cloud which is Firebase. In Firebase, it has a feature which is called ML kit.

This feature will allow developers to upload Tensorflow Lite file as model file and it will work as function API which you just send input and it will return output.

- **How we use the model in our application**

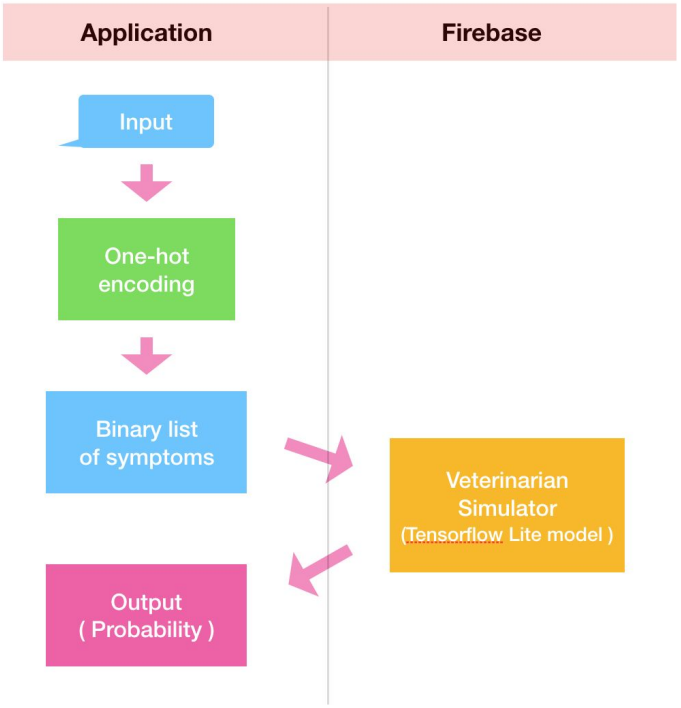


Figure 4.14 Swim Lane Diagram

As we said in the previous chapter, the selected method which we will use in this project is “one-hot & MLP”, begins with receiving an input to find the keyword by retrieve information from Firebase and compare what word is the keyword for each symptom (one-hot encoding). Then, we make an input as a binary array as input and send to the model on Firebase.

```
fun runYellowVet(input: String): Boolean {
    Log.d(TAG, msg: "runYellowVet")

    val illnessModel = CustomModelManager( modelName: "illness-classification", inputDims: 24, outputDims: 12)

    val tokenInput : Array<FloatArray> = getWordToken(input)

    if (tokenInput[0].contains(1.0f)) {
        val inputs : FirebaseModelInputs! = FirebaseModelInputs.Builder()
            .add(tokenInput) // add() as many input arrays as your model requires
            .build()

        illnessModel.firebaseInterpreter!!.run(inputs, illnessModel.inputOutputOptions!!)
            .addOnSuccessListener { result : FirebaseModelOutputs! ->
                Log.i(TAG, msg: "addOnSuccessListener")
                illnessModel.output!![0] = result.getOutput<Array<FloatArray>>(0)[0]
                val probabilities : FloatArray = illnessModel.output!![0]
                for (i : Int in 0 until illnessModel.outputDims) {
                    val current : Float = probabilities[i]
                    println("$i $current")
                }
                writeNewPost(input, probabilities, FirebaseAuth.getInstance().currentUser!!.uid)
            }
            .addOnFailureListener { e : Exception ->
                Log.e(TAG, e.message)
                Log.e(TAG, e.printStackTrace().toString())
            }
        return true
    } else {
        Log.i(TAG, msg: "Please input again in more detail")
        return false
    }
}
```

Figure 4.15 Function for using veterinarian simulator model on Firebase

As you see in the function above, we need to set the length of both input and output dimensions which are 24 and 12 respectively. To use the model on Firebase, we just call model and send set of input, after that our model will generate and return an output. After we get the output, mobile application will show a result and save to database.

- **Illness solution and information**

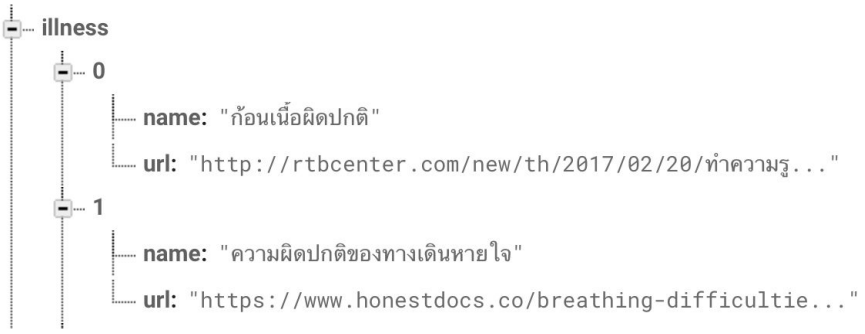


Figure 4.16 Table of illness and url information of each illness

After user get result from “Ask”, our application will allow users to click on the illness names which show on screen and it will link to information page about that illness on user’s mobile default browser.

- **Firebase Realtime Database**

It's a NoSQL cloud database that stored as JSON structure and works in asynchronous way which means every APIs calls will always return immediately without waiting for the result to continue the code so if our YellowVet users use our application more than the free limit. It will causes users can't reach our database or maybe make application unreachable and crash unless we implement callback for every call.

In terms of limits, its data stored as JSON structure so if we design a database in vertical than horizontal that means it has only one root node and make all details deep in the root node as tree structure cause the query's time of database longer to reach the child node. The solution is making multiple root node and when we want to query, we will query multiple node simultaneously and combine the completed result by some unique key.

Next, scalability, there are many popular client use Firebase Realtime Database and if YellowVet grows reach the free limit, we have to pay to scale higher limit as much as we want to. We know that it's not quite the best solution for the long term because it makes us to stick with Firebase and its owner, Google, can control their own price.

- **Users**

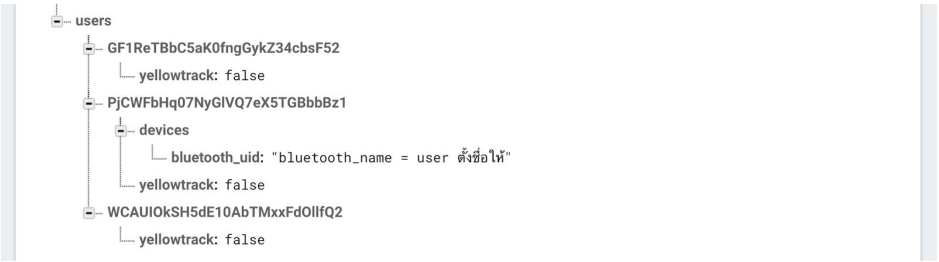


Figure 4.17 Table of all users information in YellowVet

After users sign up for the first time, it will push an initial required information such as YellowTrack devices to remember hardware.

- **Asks**

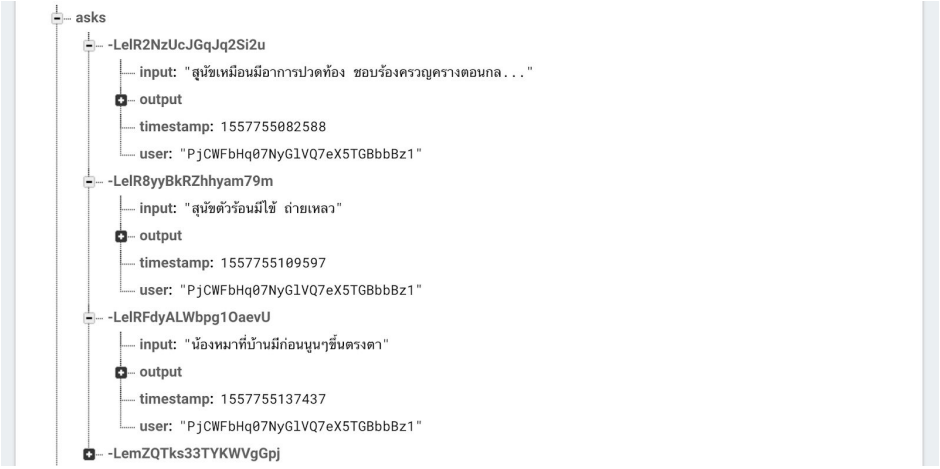


Figure 4.18 Table of all asks in YellowVet

When user add a question to the application, it will push input, output, timestamp, and especially their user unique id to database. It's very important because all questions from all members are in the same branch and separated by user unique id.

- **Google Maps API & Google Places API**

We use both Google Maps API and Google Places API for one major feature which is finding nearby veterinarian care. User can easily find the nearest veterinarian care from your location. Get basic information or direction.

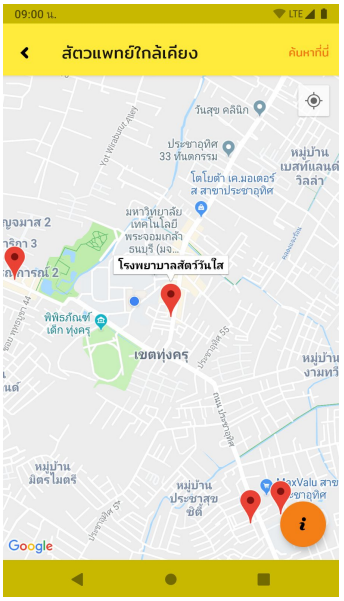


Figure 4.19 Nearby veterinarian care provided by Google

First, we have to retrieve the data from Google Maps by using Google Places API to get name, latitude, and longitude. Then, we mark all data we got and link it to Google Maps application on local devices.

However, we face some problems that Google Maps API for Android doesn't have feature to search a place by type like veterinarian care. We have to use HTTP request to Google Cloud platform and get response in JSON file. After that, we parse the file to data class and use it for our application.

After we use these APIs for a while, we found that the free version there is a limit per second that makes user request the service difficulty because of bottleneck theory.

From now on, this feature available when there are a few users online and we will decide to improve by pay a premium plan or changing API to another.

- **YellowTrack**

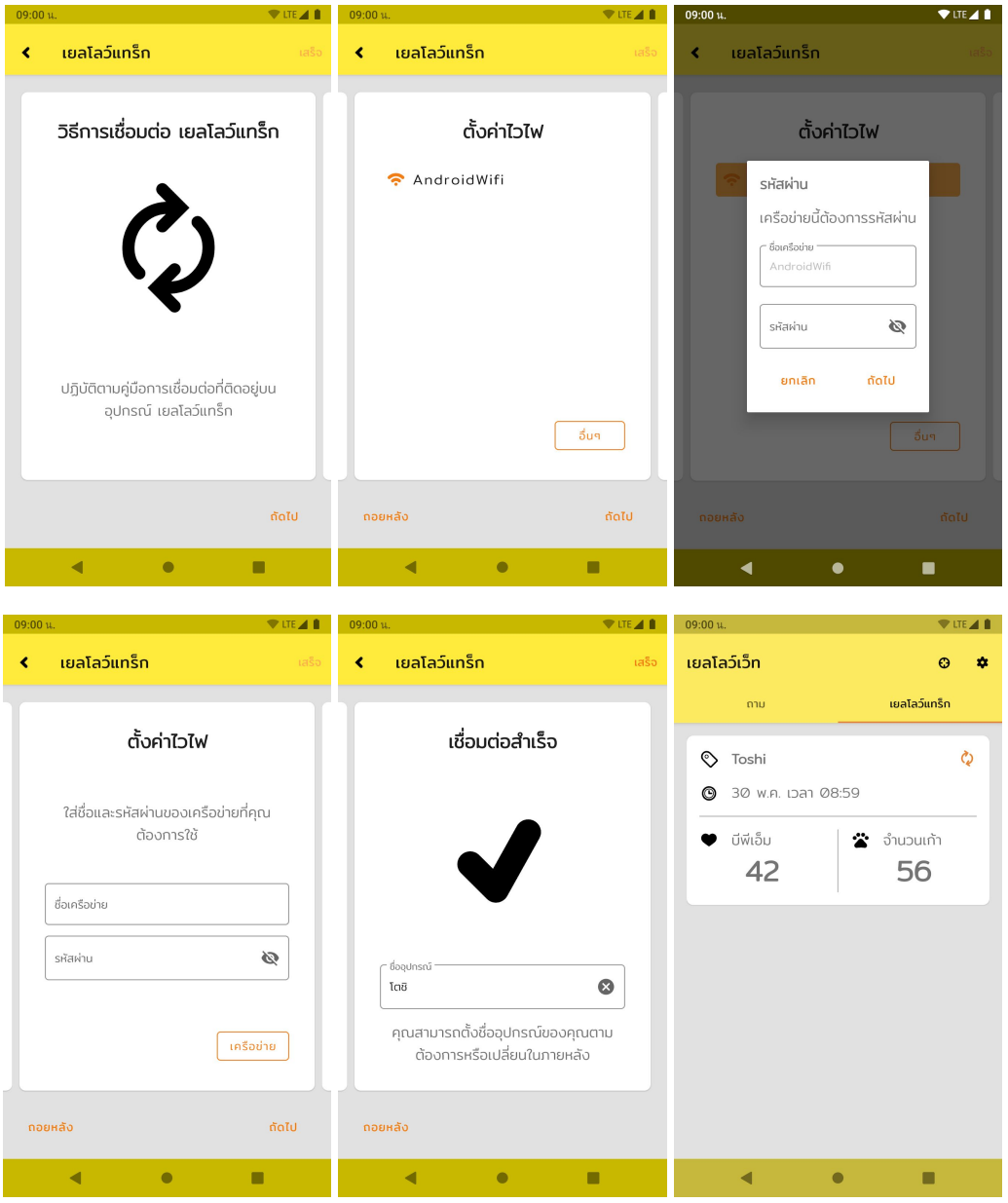
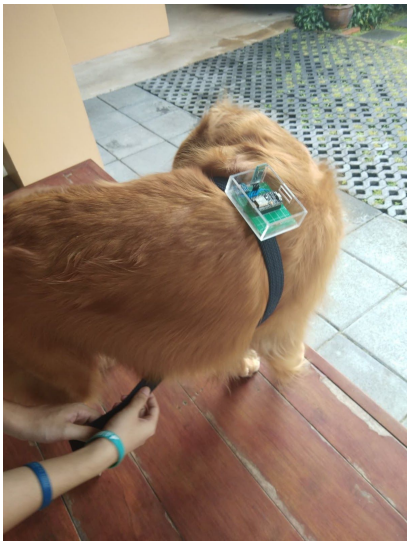


Figure 4.20 Adding YellowTrack

After we did the hardware part, we implement the simple user interface for testing the connection between them. Generally, users must pair a bluetooth between their device and YellowTrack manually first before using YellowVet application to continue config a hardware. After that, YellowVet will scan all available Wi-Fi in that area and show the result to users for convenience usage. Alternatively, users input network's name and password manually. Finally, users can change a hardware's name by whatever they want and it will show on YellowTrack's tab in home screen application.



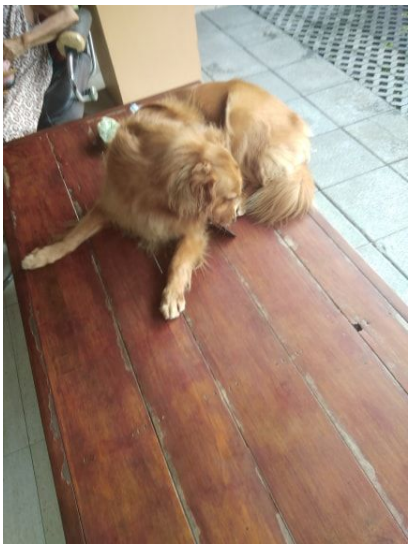


Figure 4.21 Problem from real test

We have to reduce the size of YellowVet device by using PCB for decrease unnecessary device and change the device that is smaller. Next, We got other problem are device on dog body. when we put device around stomach, Dog will try to put device out from body and stepping sensor get fault from real data.

Chapter 5

Conclusion

From our perspective, this project can satisfy objectives. It makes users feel like they have their own veterinarian which able to diagnose illness from their question or symptoms from input. Moreover, It can give an information, warning and illness which their pet might have. For this part, it increase user awareness and reduces a harm or dangerous from late treatment and receiving wrong treatment. Including case that user's pet has chance to have dangerous illness, our application will tell user to see a veterinarian as soon as possible. However, this project work as decision support or early warning application. it can't treat or heal user's pet, user still need to meet a veterinarian when our application detect illness. Our application will work like diagnosis system which learn from treatment cases in the past that came from 5-6 veterinarians. It is supervised learning machine learning model. Then, it will be more accurate when we increase training data. If we have enough data, it has chance to have diagnosis performance close as veterinarian.

5.1 Problem encountered

- **Product from multi language programming**

  The first problem which we found in this project is multiple computer languages. we decided to create an application based on the Java language because most of programming course which we take in common is Java based. Learning a new language will consume a lot of time. However, most useful library in machine learning are Python based or R based. We need to find a way to communicate between Java application and model which is created by Python. We found many architectures like AWS, Microsoft Azure, etc. but most of them are suitable for commercial or large business and some of them are not for on mobile. The last choice that we selected for this project is Google Firebase which is easy to deploy model and flexible database. Moreover, it has

features for machine learning product like ML kit which is support mobile platform.

- **Different computer (or OS) may effect to code**

  When we are following the converting model from keras to tensorflow tutorial, we found bug and many developers face a similar problem as us. This bug hold us for a long time because Tensorflow Lite is very new and it may not work in some operation system. Our solution is to use product which is called Google colaboratory. It is a free Jupyter notebook environment (Python based) that requires no setup and runs entirely in the cloud. By using google cloud environment, we can avoid bug which came from different environments.

- **Decreasing project scope**

  In our project proposal, at first time we plan to have image classification part which allow user to upload image and our application will use both image and text to diagnose the illness. However, after we research and have some experience about image processing. We think it not possible to create image classification model which can help us to diagnose the illness. Image classification need tons of training image to make it able to say “yes” or “no”'. If we need to make it able to help us diagnosis or give some useful information, we think we should create at least 5 models for each symptom and it may not be effective because it consumes a lot of time to prepare image data (labeling). Then we decided to remove this part from our project scope.

5.2 Lesson learn

Apart from academic knowledge and technical knowledge which we got from doing this project, the first thing we learned from this project is when we are designing the product we didn’t do enough the market validation. It makes our YellowVet product satisfy our expectation but it may not satisfy our target group expectations. We learned that we should validate our customer needs before make some application product. Moreover, prototype is very important, it make us know what is problem of our application. It is faster and easier than develop real application and get feedback. If we

make product and not listen to other people, we may stuck in our perspective and make our application has some mistakes.

The second thing that we learn is understanding selected tool, as an engineering student we should know the advantages and disadvantages between tools which we will use. We should not use it as black box.We should compare it for selecting the best "Trade-off" that suit with our project situation or our goal. We should design experiments and create assumption for each experiment and make discussion to answer why it work or not work.

5.3 Future work

After we finished making application, we present it to veterinarians and gather their comments. According to advice from veterinarian, To prevent dangerous from using YellowVet, our application should have warning or feature which can prove that the output result came from real veterinarian. In the next version, we plan to have a new feature called "Question Similar Matching". This feature will be addition part of "Ask" feature. It will show questions in past which similar to user's question. It will allow users to click and view what is that question and what is the answer from veterinarian. This feature will give users more information and understand other treatment case study.

Moreover, we also provide some privacy policy or term of agreement before users use our YellowVet application. Everytime new users register, they must agree with these policy before use our application.

The next thing is improving the YellowTrack to support diagnosis. In our current progress we just set condition by ourselves. From our perspective, we should do some research about abnormal behavior in pet for writing condition or threshold. It will consume a lot of time but it will make our diagnosis more accurate. Moreover, YellowTrack may have more function apart from step tracking and breathing sensor.

The last plan is to make this application work with many popular pets such as cats, birds, fish, etc. It will consume a lot of time to make it work. It may be better if we spent time to develop this application for dog only. However, making YellowVet is usable for any kind of pet is what we want to do for the first time.

Table 5.1 Completion status of each component of project

Component	Status
Machine Learning	
- Natural Language Processing Experiment	Complete
- Machine Learning Experiment	Complete
- Export model	Complete
- Put on Firebase's ML Kit	Complete
- Image Classification	Not Started
Application	
- Connect with Veterinary simulator	Complete
- Allow to input and return output	Complete
- User interfaces	Complete
- Deploy application to Play Store	Complete
- Connect with IoT	Partially Complete
- Illness information page	Partially Complete
- Map API	Complete
Internet of Things	
- Test sensor	Complete
- Test function for tracking sensor	Complete
- Test function for breathing sensor	Complete
- Connect to Firebase	Complete
- Connect with mobile app	Partially Complete
- Prototype & Testing	Partially Complete

References

1. ทีมข่าวสด, 2018, **ธุรกิจสัตว์เลี้ยงแตะ 3.2 หมื่นล้าน รับแนวโน้มคนโสด-สูงอายุฟุ้ง** [Online], Available: https://www.khaosod.co.th/economics/news_1066457 [2018, September 28].
2. Chatchapong Mekgamol, 2017, **Kotlin คืออะไร และทำไมต้อง Kotlin** [Online], Available: https://medium.com/imkrz/kotlin-คืออะไร-และทำไมต้อง-kotlin-548a84ca4cf [2017, May 29]
3. JosveBot, 2018, **Firebase** [Online], Available: https://en.wikipedia.org/wiki/Firebase [2018, September 29]
4. Thatchaphon Kaeosuriya, 2017, **รีวิว Deep Learning Library Episode 1 : Keras** [Online], Available: https://medium.com/@thatchaphonkaeosuriya/รีวิว-deep-learning-library-episode-1-keras-557e2b6a66bf [2017, June 10]
5. mk,2018,**รู้จัก TensorFlow Lite เอนจินสำหรับ Deep learning บนมือถือ ที่นักพัฒนาไม่มีทางหนีพ้น** [Online], Available: https://www.blognone.com/node/102192 [2018, May 11]
6. Faisol Hayee, 2018, **3SB04: Android Development** [Online], Available: https://medium.com/@faisolhayee/3sb04-android-development-a090fe92f3c4 [2018, February 18]
7. The Raspberry Pi Foundation, 2018, **Raspberry Pi 3 Model B** [Online], Available: https://www.raspberrypi.org/products/raspberry-pi-3-model-b
8. Adan Aqeel, 2018, **Introduction to Arduino Uno**, Available: https://www.theengineeringprojects.com/2018/06/introduction-to-arduino-uno.html [2018, June 21]
9. Armtronix, 2016, **Steps to Setup Arduino IDE for NodeMCU ESP8266**, Available: https://www.instructables.com/id/Steps-to-Setup-Arduino-IDE-for-NODEMCU-ESP8266-WiF
10. Lex C,2017,**Basics: Project 031b ADXL345 (GY-291) three-axis accelerometer module**, Available: http://acoptex.com/project/231/basics-project-031b-adxl345-gy-291-three-axis-accelerometer-module-at-lex-c/#sthash.op63Kwdl.dpbs [2017, December 5]

11. 2017, **Connect Android device to Arduino via HC-06 Bluetooth module**, Available: http://androidsmile.com/index.php/android/connect-android-to-arduino [2017, January 19]
12. chang, 2018, **แนะนำบอร์ด Lamloei 32 Lite กับ การใช้งานบน Arduino แบบเบื้องต้น,** Available: http://www.ayarafun.com/2018/12/how-to-setup-lamloei-32-lite-with-arduino [2018, December 19]
13. Manning Ch. D., H. Schütze, 1999, **Foundations of Statistical Natural Language Processing**. Cambridge, MA: The MIT Press.
14. lew, 2017, **Data Scientist จาก True สร้างระบบตัดคำแบบ Deep Learning ด้วย Keras เปิดซอร์สแบบ MIT** [Online], Available: https://www.blognone.com/node/93500 [2018, September 30]
15. wannaphong, 2017, **PyThaiNLP - โมดูล NLP ภาษาไทยใน Python** [Online], Available: https://python3.wannaphong.com/2017/05/pythainlp-nlp-python.html [2018, September 30]
16. Athif Shaffy , 2017, **Vector Representations of Text for Machine Learning** [Online], Available : https://medium.com/@athif.shaffy/one-hot-encoding-of-text-b69124bef0a7 [2017, November 8]
17. Bartosz Góralewicz, 2018, **The TF*IDF Algorithm Explained** [Online] , Available : https://www.onely.com/blog/what-is-tf-idf/ [2018, March 26]
18. NSS, 2017, **An Intuitive Understanding of Word Embeddings: From Count Vectors to Word2Vec** [Online] , Available : https://www.analyticsvidhya.com/blog/2017/06/word-embeddings-count-word2veec/ [2017, June 4]
19. Gustavo Chávez , 2019 , **Implementing a Naive Bayes classifier for text categorization in five steps** [Online] , Available : https://towardsdatascience.com/implementing-a-naive-bayes-classifier-for-text-categorization-in-five-steps-f9192cdd54c3 [2019, February 28]
20. Nithin Kumar K ain , 2018, **Understanding of Multilayer perceptron (MLP)** [Online] , Available: https://medium.com/@AI_with_Kain/understanding-of-multilayer-perceptron-mlp-8f179c4a135f [2018, November 22]

21. Colah, 2015, **Understanding LSTM Network** [Online] , Available: https://colah.github.io/posts/2015-08-Understanding-LSTMs/ [2015, August 27]

22. Your.md, 2013, **Your MD. OneStop health** [Online], Available: https://www.your.md/

23. Sensely, 2013, **Sensely - How are you feeling** [Online], Available: https://www.sensely.com/

24. OSDCO, 2018, **พูดคุย สอบถาม ปรึกษาปัญหา ของสัตว์เลี้ยงกับสัตวแพทย์ผู้เชี่ยวชาญจากทีม OSDCO** [Online], Available: https://www.osdco.net/consulting/

25 Firebase, 2019, **Privacy and Security in Firebase** [Online], Available: https://firebase.google.com/support/privacy?fbclid=IwAR0PzjvaeJnLxAuyIZCY_d0ErXLwM0NrB68BHvVZ93f8fdBmKBeGNNFDKAY [2019, June 6]