**TABLE OF CONTENTS**

[LIST OF FIGURES 3](#_Toc18802087)

[LIST OF TABLES 4](#_Toc18802088)

[ABBREVIATION LIST 5](#_Toc18802089)

[Chapter 1: INTRODUCTION 6](#_Toc18802090)

[1.1. Reasons 6](#_Toc18802091)

[1.2. Objectives 6](#_Toc18802092)

[1.3. Scope 6](#_Toc18802093)

[1.4. Related work 7](#_Toc18802094)

[Chapter 2: THEORY 8](#_Toc18802095)

[2.1. Machine learning 8](#_Toc18802096)

[2.2. TensorFlow and TensorFlow Lite introduction 9](#_Toc18802097)

[2.3. Raspberry Pi and supporting devices 10](#_Toc18802098)

[2.4. Flutter 13](#_Toc18802099)

[2.5. Firebase 14](#_Toc18802100)

[2.6. NoSQL database model 14](#_Toc18802101)

[Chapter 3: EXPERIMENT 16](#_Toc18802102)

[Section 1: Trash bin model 16](#_Toc18802103)

[3.1. Workflow 16](#_Toc18802104)

[3.2. Implementation 17](#_Toc18802105)

[3.3. Accuracy evaluation 22](#_Toc18802106)

[3.4. Experimental results 26](#_Toc18802107)

[Section 2: Mobile application 26](#_Toc18802108)

[3.1. Overview 26](#_Toc18802109)

[3.2. Design 27](#_Toc18802110)

[3.3. Database design 28](#_Toc18802111)

[3.4. Functions 29](#_Toc18802112)

[Section 3: Whole system 36](#_Toc18802113)

[3.1. System Architecture 36](#_Toc18802114)

[3.2. Status detection 37](#_Toc18802115)

[3.3. Connection building 37](#_Toc18802116)

[3.4. Data Updating 37](#_Toc18802117)

[Chapter 4: SUMMARY 39](#_Toc18802118)

[4.1. Summary 39](#_Toc18802119)

[4.2. Development orientation 39](#_Toc18802120)

[REFERENCES 41](#_Toc18802121)

# LIST OF FIGURES

[Figure 1. Raspberry Pi 11](#_Toc18802043)

[Figure 2. Raspberry PI camera module V1 5MP 12](#_Toc18802044)

[Figure 3. PIR sensor 12](#_Toc18802045)

[Figure 4. Servo MG90S 13](#_Toc18802046)

[Figure 5. Processing flow 16](#_Toc18802047)

[Figure 6. Algorithm testing result 23](#_Toc18802048)

[Figure 7. Firestore’s document storage 27](#_Toc18802049)

[Figure 8. Collection model 28](#_Toc18802050)

[Figure 9. Get user ID function workflow 30](#_Toc18802051)

[Figure 10. Listing function workflow 31](#_Toc18802052)

[Figure 11. Adding trash bin function workflow 32](#_Toc18802053)

[Figure 12. Showing information function parameter workflow 33](#_Toc18802054)

[Figure 13. Adding and editing information functions workflow 34](#_Toc18802055)

[Figure 14. Removing trash bin function workflow 35](#_Toc18802056)

[Figure 15. Notifying function workflow 36](#_Toc18802057)

[Figure 16. System architecture 36](#_Toc18802058)

[Figure 17. Experiment picture 39](#_Toc18802059)

# LIST OF TABLES

[Table 1. Database dictionary 28](#_Toc18802074)

[Table 2. Get user ID method 29](#_Toc18802075)

[Table 3. Listing function parameter 30](#_Toc18802076)

[Table 4. Listing function method 30](#_Toc18802077)

[Table 5. Adding trash bin function parameter 31](#_Toc18802078)

[Table 6. Adding trash bin function parameter 31](#_Toc18802079)

[Table 7. Showing information function parameter 32](#_Toc18802080)

[Table 8. Showing information function method 32](#_Toc18802081)

[Table 9. Adding and editing information functions parameter 33](#_Toc18802082)

[Table 10. Adding and editing information functions method 33](#_Toc18802083)

[Table 11. Removing trash bin function parameter 34](#_Toc18802084)

[Table 12. Removing trash bin function method 34](#_Toc18802085)

[Table 13. Notifying function method 35](#_Toc18802086)

# ABBREVIATION LIST

|  |  |  |
| --- | --- | --- |
| **No.** | **Abbreviation** | **Origin word** |
| 1 | AI | Artificial intelligent |
| 2 | CNN | Convolutional neural network |
| 3 | FCM | Firebase Cloud Messaging |
| 4 | IoT | Internet of Things |
| 5 | ML | Machine learning |
| 6 | PIR | Passive infrared sensor |
| 7 | SVM | Support vector machines |

# INTRODUCTION

The research transforms a normal trash bin into a “smarter” one by applying computer vision technology. With the support of sensor devices, the trash bin can automatically classify garbage. In details, a camera on the trash bin takes pictures of trash, then the central processing unit analyses and makes decisions which bin to drop trash into. The accuracy of this process reaches 86% theoretically.

Besides, this bin model is also connected to the Internet in order to upload the status (normal or full) on the Internet for further management. A mobile application is also developed to let users track the status of the trash bin as well as provides some information management such as adding the trash bin they have and gives some notes for easier uses.

This chapter will provide an overview for this research.

### Reasons

Waste management has attained the great concern. People are aware of the impact of improper waste disposal practices, but the inconvenience of implementation leads to unexpected results. In Vietnam, the Decision No.44/2018/QD-UBND has required Vietnamese citizens to perform waste classification at home. Currently, the progress is not satisfactory. Therefore, this project will build a system to automatically classify domestic rubbish at a reasonable price to help manage trash, protect environment and support communities.

### Objectives

The objective of the project is to implement a smart domestic waste classification system. Its main product is a trash bin which is able to automatically classify garbage as well as connect with a mobile application to allow user to perform management operations.

In order to do that, we developed a garbage classification algorithm running on a Raspberry Pi. That device is responsible for connecting with sensors, which allows the trash bin to operate automatically. Besides, considered as a mini computer, a Raspberry Pi device with appropriate configuration can send data to the mobile application, which fully meets the requirement of this research.

Another important part of this system is a mobile application for user to manage information and track the status of the trash bin they own. This application is developed with Flutter framework and Firebase platform for back-end development.

### Scope

Within the scope of this research, the main content is developing a basic system allows both trash bin and mobile application work well together. Therefore, the research progress has an assumption on some default information of the trash bin (serial number and corresponding key) is setup on database directly instead of using another mediate application to insert these data. The whole application also works in ideal conditions with stable Internet connection condition and no invalid data input, which allows us to focus on developing the main functions. Internet connection checking and data validation will be finalized before practical launch of this system.

### Related work

In such a modern and busy life with strong industrial development, smart and automated applications are being developed to provide time-saving and convenience for everyone. In terms of waste management, which has attained the great concern of the world, there have been several studies aimed at developing automatical waste classification products through the use of modern technology.

In a research named “SMART TRASH CAN USING IOT” [1], the authors aimed to solve the problem of waste in India by creating an IoT application trash bin. They connected the bin to the Internet and track the status of the bin by using Raspberry Pi devices and other sensors such as infrared sensor (for motion detection), microcontroller PIC16F876A, humidity sensor and ultrasonic sensor. Using these devices, they provide simple waste sorting solution: garbage moisture and dry garbage. The study resulted in up to 90% accuracy for dry waste and 88% for wet waste. Another project on automatic trash classification [2]used electronic and sensors knowledge to build an automated waste sorting system with cost-saving and easy using for family. In details, the product in this study is designed to classify metal, moisture and dry waste using a parallel resonance impedance mechanism to detect metals and capacitive sensor for wet and dry waste.

In addition, artificial intelligence has been also applied in garbage classification to provide better efficiency. For example, computer vision technology was used on collected garbage images to classify into six categories including glass, paper, metal, plastic, cardboard and other waste [3] Gary Thung and Mindy Yang, the authors manually collected data containing about 400-500 images for each class and published this dataset. Their research used Support Vector Machines (SVM) models and Convolutional Neural Networks (CNN) and that experiments showed a result that SVM outperformed CNN. Another study of computer vision application used Thung and Yang’s database [4]and tested different methods to find the best performance among Pre-trained models VGG-16, AlexNet, SVM, K-Nearest Neighbor (KNN) and Random Forest (RF). Their progress gave a result of up to 93% accuracy with the VGG-16 model [5]Based on these researches, we conduct our own project to build a trash bin that can automatically classify garbage an additional mobile application for users to monitor and track the information of the trash bin.

# THEORY

In this section, we introduce the theoretical foundation for this research such as Machine learning, its applications and types. We also provide a basic information on TensorFlow, TensorFlow Lite and related definitions. In addition, the description of other electrical devices evolved in the research also included here: Raspberry Pi V3 B+, camera module, MG90S servo motor, PIR sensor (PIR HC-SR501) and IR sensor. For the mobile application, Flutter framework which uses Dart language and Firebase platform are the foundation for our development.

### Machine learning

#### Definition

Machine learning is a subfield of Artificial Intelligence (AI) field, which uses algorithms that allow computers to learn from data without being explicitly programmed.

#### Applications

Image processing is to solve the problems of analyzing information from images or performing some transformations. There are some big achievements applied this technique such as Photo Tagging[[1]](#footnote-1) of Facebook, Optical Character Recognition[[2]](#footnote-2) or self-driving cars of Google[[3]](#footnote-3) or Tesla[[4]](#footnote-4) . This application is the main application used in this research.

Besides, there are other applications such as text analytics, data mining and video games & robots

#### Classification

In general, there are two basic classifications of machine learning algorithms including supervised machine learning and unsupervised machine learning. The main differences between them are how we provide the dataset for the model, how the algorithm uses those data and the type of problem to be solved. Besides, machine learning can be also classified into reinforcement learning and semi-supervised learning [3]Supervised learning

In supervised learning, the machine learning algorithm performs the process of predicting the output (label) of one of the newly input data based on the previously trained dataset (training set). Each sample of data in the training set is called training example. Machine learning algorithms need to learn how to convert each input into corresponding results. Therefore, any sample of training data in the training data set needs to know its label in advance.

Supervised learning is the most common group in machine learning algorithms. This group is also subdivided into two other groups: Regression when the output desires a continuous value and classification when our desired output is a finite and discrete set.

##### Unsupervised learning

In unsupervised learning data samples only require input without output (label). The unsupervised learning algorithm will rely on the structure of the data to perform a certain task, such as clustering or reducing the dimension for more convenience in storage and calculation.

Unsupervised learning problems can be further divided into two categories: Clustering all data into small groups based on the association between the data in each group; Association uses to discover a rule based on a lot of given data.

##### Semi-supervised learning

Semi-supervised learning is a problem when we have a large amount of data but only a part of them is labeled.

##### Reinforcement learning

Reinforcement learning is a problem that helps a system to automatically determine context-based behavior to achieve the highest performance. This method is mainly applied to Game Theory (Game Theory), the typical example is AlphaGo by Google.

### TensorFlow and TensorFlow Lite introduction

#### TensorFlow library

“TensorFlow is an open source software library for numerical computation using data flow graphs. Originally developed by researchers and engineers working on the Google Brain Team within Google's Machine Intelligence research organization for the purposes of conducting machine learning and deep neural networks research”[[5]](#footnote-5)

In other words, TensorFlow is an open source library for machine learning developed by Google. It built a comprehensive and flexible tools, libraries and community; therefore, it is widely used in machine learning applications nowadays.

#### TensorFlow Lite

TensorFlow Lite[[6]](#footnote-6) is a compact version of TensorFlow. It is a tool specifically developed for developers on mobile devices and IoT. Its advantage is that it can run fast on low latency devices and especially, require less capacity on those devices.

TensorFlow Lite trains AI models by using TensorFlow tools, then the models will be converted to Lite version to be able to operate on mobile devices.

#### Some basic concepts in TensorFlow and TensorFlow Lite

There are 4 main concepts [6]

##### TensorF

TensorFlow provides a kind data called Tensor, which is the most basic concept in TensorFlow. All data in TensorFlow are Tensor, so the exchange of data during processing only through Tensor. In different words, Tensor is a multidimensional array with several other reference attributes attached [7]. The properties of Tensor described in the document include:

* Device: the current name of the device that the Tensor will be published. It can be assigned as ‘none’.
* Graph: graph containing the current Tensor.
* Name: name of the current Tensor.
* Shape: return TensorShape describe the Shape of the current Tensor.
* Op (operation): operation which is used to publish the current Tensor.
* Dtype: types of elements in the current Tensor.

##### Rank

Rank is degree or depth of a Tensor. For example, Tensor = [1] will have a rank of 1, Tensor = [[[1,1,1], [178,62,74]]] will have a rank of 3, Tensor = [[1,1,1], [178,62,74]] will have a rank of 2. The method to determine the rank of a Tensor is to count the number of opening brackets to the value other than the first square bracket. This ranking is important because it helps to classify Tensor's data. When in a particular rank, Tensor has its own names as follows:

* Scalar: when Tensor has a rank of 0, Tensor represents a specific number or sequence. Example: scalar = 123.
* Vector: vector is a Tensor rank 1. In python, Vector is a list or a one-dimensional array [Python] contains numbers. For example: list = [123,345].
* Matrix: this is a rank 2 Tensor or a two-dimensional Python concept. Example: matrix = [[1,2], [2,1]].
* N-Tensor: when Tensor rank is greater than 2, it is called N-Tensor.

##### Shape

Shape is a tuple in Python that has dimensions. The number of dimensions is equal to the rank of the corresponding Tensor used to describe that Tensor’s structure.

##### Type

Type is data type of elements in Tensor. A Tensor has only one Type attribute, so it is inferred that there is only one Type for all elements present in the current Tensor.

### Raspberry Pi and supporting devices

In this research, 4 main devices were used: Raspberry Pi 3 B+[[7]](#footnote-7), Raspberry Pi camera V1 5MP module[[8]](#footnote-8), servo motor MG90S[[9]](#footnote-9) and PIR sensor SR501 module[[10]](#footnote-10).

#### Raspberry Pi

Raspberry Pi (Figure 1) is considered to be a small-sized computer with powerful built-in hardware. It is capable of running the operating system and many applications on it. At first, Raspberry Pi Foundation organization developed Raspberry projects for teaching purposes, however, after that Raspberry Pi devices have gained positive evaluations and become popular until now.

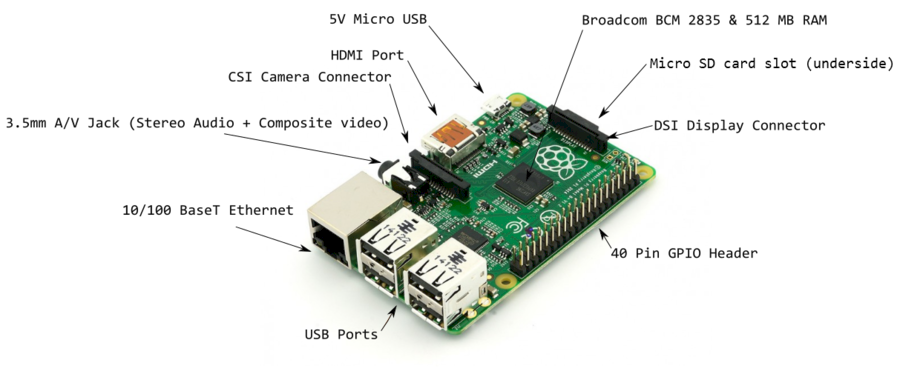


Figure 1. Raspberry Pi

Source: http://mlab.vn/11025-raspberry-pi-la-gi-gioi-thieu-cac-ung-dung-cua-raspberry-pi-3.html

Raspberry Pi 3 Model B+ is a processor and network connector. Model B+ uses Broadcom BCM2837B0 4 1.4GHz processor. Pi 3 Model B+ can respond to Wi-Fi 2 bands: 2.4GHz and 5GHz Gigabit Ethernet (via USB 2.0 port) speeds up to 300Mbps for tasks that require fast network speeds. This device also supports Bluetooth 4.2 and Bluetooth LE for better connection with other smart devices.

Specification about Raspberry Pi 3 Model B+:

* SoC: Broadcom BCM287B0, Cortex-A53(ARMv8) 64-bit SoC @ 1.4 GHz
* RAM: 1GB LPDDR2 SDRAM
* Wi-Fi b/g/n/ac
* Bluetooth 4.2, BLE
* Gigabit Ethernet over USB 2.0(maximum throughput 300Mbps)
* 40-pin GPIO
* HDMI
* 4 USB 2.0 ports
* Micro SD port for loading your operating system and storing data
* Power-over-Ethernet (PoE) support (requires separate PoE HAT)

Raspberry Pi played an important role in the whole study.

#### Raspberry PI camera module V1 5MP

The Raspberry Pi camera acted as the eyes of the entire system by transmitting images taken from daily waste to the Raspberry Pi to analyze.

****

Figure 2. Raspberry PI camera module V1 5MP

Source: https://robu.in/product/raspberry-pi-camera-module/

Raspberry Pi camera V1 5 Megapixel has high light sensitivity and works well in many different lighting conditions (both indoor and outdoor conditions). A special feature that the camera offers is high-definition shooting during movie shooting.

Raspberry Pi has a CSI socket slot for attaching cameras instead of USB ports. This helps limit bandwidth congestion for USB processing chips on Raspberry Pi. The length of the camera cable is sufficient to ensure the speed of transferring images from the module to the Raspberry Pi. Before using the Raspberry Pi camera, it is required to set up the system configuration to activate the camera on Raspberry Pi.

#### PIR Sensor

PIR is a sensor connected by 2 words: Passive and InfraRed. “Passive” indicates that the sensor does not actively emit the IR signals itself, but passively detects the IR radiations coming from other objects. The normal human body temperature range is typically stated as 36.5–37.5 °C, so it can detect the infrared radiation coming from the human body moving into the surrounding area where the sensor is placed. The sensor can be adjusted for sensitivity by two variable resistors: Sx resistors (adjust the sensitivity) and Tx resistor (adjusts the sensor's closing time).

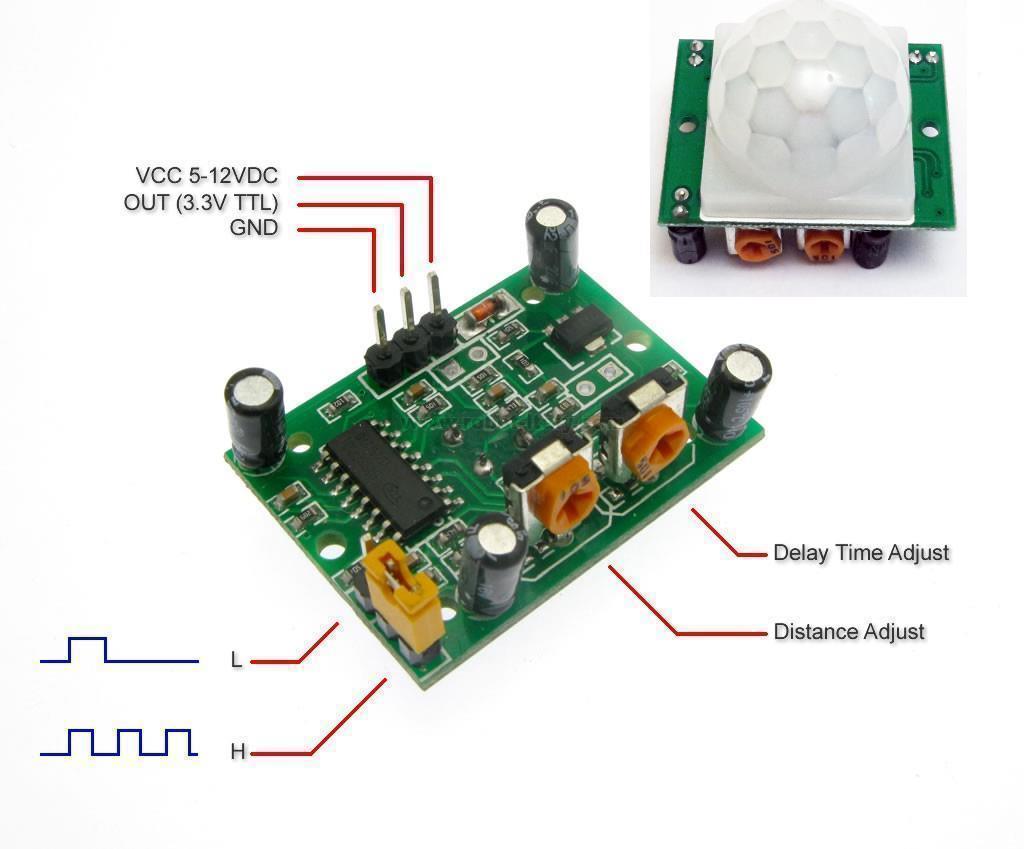
****

Figure 3. PIR sensor

Source: http://www.dientuspider.com/san-pham/chuyen-dong-rung/cam-bien-chuyen-dong-pir-hc-sr501-79.html

Moreover, the PIR also has a hemispherical glass which is called Fresnel. Fresnel helps to prevent ultraviolet rays while extending the detection angle to the PIR. In this research, PIR is used to sense the motion of surrounding people and activate the camera as well as the processor if it senses a person moving.

Features and characteristics:

* Working voltage range: DC 4.5V- 20V
* Output voltage: signal high / low: 0-3.3V output
* Working temperature: -20- + 80 ° C
* Activation method: repeated activation L / H enabled repeat
* Time delay: 5-200S (adjustable, default 5s + -3%)
* Scope behavior detection: <140 °
* Using sensor: 500BP
* Gap detection: 3--7M (adjustable)
* Dimensions: 3.2cm x 2.8cm x 2,4cm (approximate)

#### RC Servo MG90S Motor

RC Servo MG90S is a micro servo motor with metal gear. This small and lightweight servo comes with high output power, thus ideal for strong traction and high durability. RC Servo motors are controlled like the common ones on the market. The motor is used for moving waste to the right place through the steps of programming control.

****

Figure 4. Servo MG90S

Source: https://market.samm.com/mg90-mikro-servo-motor-metal-disli

Features and characteristics:

* Product Name: MG90S servo
* Weight: approximately 13,4g
* Size: 22.8 x 12.2 x 28,5mm
* Torque stability: 1.8kg / cm (4,8V), 2.2 kg / cm (6V)
* Operation Speed: 0.1 seconds / 60 degree (4,8v), 0.08 seconds / 60 degree (6V)
* Operating voltage: 4, 8-6,0V
* Length wire connection: 175mm)
* Gear type: Metal.

### Flutter

Flutter is cross-platform which can develop native interface that is compatible with Android and IOS simultaneously. It uses Dart language developed by google and offers lots of widgets in order to create user interfaces following Material Design principles.

Flutter properties[[11]](#footnote-11)

* Fast Development: It can paint the app to life in milliseconds with Stateful Hot Reload. Its rich set of fully-customizable widgets to build native interfaces in minutes.
* Expressive and Flexible UI: Flutter allows quickly shipping features with a focus on native end-user experiences. Its layered architecture allows for full customization, which results in incredibly fast rendering and expressive and flexible designs.
* Native Performance: Flutter’s widgets incorporate all critical platform differences such as scrolling, navigation, icons and fonts to provide full native performance on both iOS and Android.

##### Dart

Dart[[12]](#footnote-12) is a programming language used in Flutter for frontend development from Google. It is easy, productive, fast, portable with C-style syntax and can also be used for web, servers, desktops.

Dart is indispensable for Flutter with its features: AOT (Ahead of Time), JIT (Just in Time). Dark makes it easier to create smooth animations and transitions that run at 60fps, as well as avoids preemptive scheduling and shared memory (and thus locks).[[13]](#footnote-13)

##### Material design

Material Design[[14]](#footnote-14) is the design that was developed by Google. It is inspired by the real physical world and real textures. Therefore, it can make user feel comfortable and give them a sense of reality. For example, it can express reflection of light, cast of shadow and so more.

In addition, it is cross-platform design so it can be shared across the platform (for example, it can be used in both Android and iOS operating system)

### Firebase

Firebase[[15]](#footnote-15) is Google's mobile platform that helps you quickly develop high-quality apps. It is a Backend-as-a-Service (BaaS) app development platform that provides hosted backend services such as a real-time database, cloud storage, authentication, crash reporting, machine learning, remote configuration, hosting for your static files and so on. All of them supports Flutter.

With Firebase, we can shorten the development and deployment. It also supports both Android and IOS foundation. In this research, the services for Cloud Firestore database, Cloud Messaging and Cloud Functions are used to develop our mobile application, which will be described in more details at Experiment chapter.

### NoSQL database model

NoSQL is a non-relational database management system without a fixed schema. It is is used for distributed data stores with humongous data storage needs. NoSQL is best suitable for Big data and real-time applications.

NoSQL databases are usually categorized under one (or more) of the following types[[16]](#footnote-16)

**Document databases** pair each key with a complex data structure known as a document. Documents can contain many different key-value pairs, or key-array pairs, or even nested documents.

**Graph stores** are used to store information about networks of data, such as social connections. Graph stores include Neo4J and Giraph.

**Key-value stores** are the simplest NoSQL databases. Every single item in the database is stored as an attribute name (or 'key'), together with its value. Examples of key-value stores are Riak and Berkeley DB. Some key-value stores, such as Redis, allow each value to have a type, such as 'integer', which adds functionality.

**Wide-column** stores such as Cassandra and HBase are optimized for queries over large datasets, and store columns of data together, instead of rows.

# EXPERIMENT

This section describes the overall working progress and results of the system developed. The classification process, sensor implementation, trash bin design as well as mobile application development are all presented in details.

## Trash bin model

### Workflow

We divided garbage into 2 main categories: recyclable and non-recyclable trash. This classification method can be changed to serve different agencies or countries depending on their classification standards. However, there is a limitation of 2 main categories for the design of this trash bin model developed in this research to work accurately and smoothly (Figure 5).

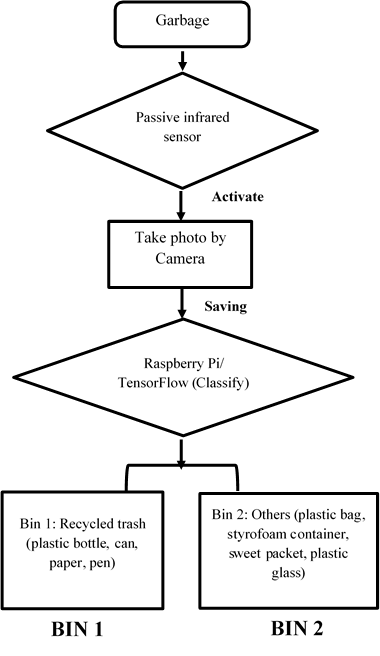


Figure 5. Processing flow

The processing flow contains 4 main steps:

**Step 1:** First of all, user puts trash in a temporary garbage container. The PIR sensor receives the PIR signal and activates the system.

**Step 2:** The camera is activated for capturing the trash and store temporarily in Raspberry Pi storage.

**Step 3:** Raspberry Pi (TensorFlow Lite) uses the captured image to analyze.

**Step 4:** Immediately after getting the classifying result, if the trash is classified into recyclable trash such as paper, can, plastic bottle or pen, it will be put into bin 1. If the result is not recyclable trash such as plastic bag, plastic glass, sweet packet or styrofoam container, they will be put into bin 2.

### Implementation

#### Collecting dataset

First of all, while training a model for image classification application, the image used will greatly affect the results. The number of images and their diversity are the two most important factors to ensure that the model can reach high accuracy. In order to prepare this dataset, we collected images from 2 sources:

###### Data from the Internet

A large amount of data collected by using Google images includes: styrofoam boxes, cans, plastic bottles, plastic glasses, food packets, plastic bags, papers and pens. Those images had good quality with white background. Total amount of image collected by this method is 4700 pictures with approximately 400-700 images for each category.

Moreover, we also used the dataset provided by Gary Thung and Mindy Yang from their Classification of Trash for Recyclability Status[9] project, containing totally 2527 images of glass, paper, cover carton, plastic, metal and some other types of garbage in the condition of white background and normal light (sunlight or room lights).

###### Data collected manually

In order to increase the diversity for the dataset, we have also taken daily waste pictures, especially some categories commonly found in schools such as tissues, styrofoam boxes, food packets and so on. However, the time has been quite short since we started to progress and the number of our members is limited, so the amount of data collected is still quite small. Therefore, data from this source should be collected more to be able to play a role in this research.

#### Eliminate error images:

This step was done manually by removing blank or error images.

The following preparing, training, fine tuning and converting model to TensorFlow Lite model were based on the instructions provided by TensorFlow development team on Google Colab[[17]](#footnote-17)

#### Preparing

Data needs to be processed using the following library before any training:

##### Step 1: Importing necessary library

unicode\_literals library: allows us to represent binary and string symbols in ascii code with a simpler syntax.

print\_function library: allows us to use print statement as a function.

absolute\_import library: allows us to load modules (module names) directly.

division library: requires all modules when performing division to return the approximate value since some classic division returns the rounded integer result.

os library: finds and retrieves the address of the directory containing the data set

TensorFlow library: performs tasks such as dividing dataset into training and validation dataset, grouping data into batches, resize input data, defining model, training models, and predicting.

##### Step 2: Importing dataset

Data can be loaded via the command: base\_dir = os.path.join (os.path.dirname (directory), \* path)

os.path.dirname (*directory*) returns the parent path of the *directory*

os.path.join() combines paths and returns the result as a path.

from zipfile import ZipFile

with ZipFile(datasetFile, 'r') as zipObj:

# Extract all the contents of zip file in different directory

zipObj.extractall('waste')

base\_dir = os.path.join(os.path.dirname('waste'), 'waste/DataSet')

##### Step 3: Dividing the data set into training validation data set and modifying

In this step, TensorFlow.keras.preprocessing.image.ImageDataGenerator() and flow\_from\_directory() functions were used

TensorFlow.keras.preprocessing.image.ImageDataGenerator(): allows us to standardize dataset input (range 0-255 to 0-1) and determine data rates for training and validation.

TensorFlow.keras.preprocessing.image.ImageDataGenerator().flow\_from\_directory(): allows us to determine the input data size for the model and determine the value for each batch.

IMAGE\_SIZE = 224

BATCH\_SIZE = 64

datagen = tf.keras.preprocessing.image.ImageDataGenerator(

rescale=1./255,

validation\_split=0.2)

train\_generator = datagen.flow\_from\_directory(

base\_dir,

target\_size=(IMAGE\_SIZE, IMAGE\_SIZE),

batch\_size=BATCH\_SIZE,

subset='training')

val\_generator = datagen.flow\_from\_directory(

base\_dir,

target\_size=(IMAGE\_SIZE, IMAGE\_SIZE),

batch\_size=BATCH\_SIZE,

#### Training model

This training process was based on transfer learning techniques. With large-dataset problems, it is rare to build and re-train the entire model from scratch since there are few suitable datasets. Instead, the most commonly used method is to use pre-trained models, which is called as transfer learning, to create good enough model with small dataset by using available models.

This step was done by instantiating the pre-trained model and adding a fully-connected classifier on top. The pre-trained model was "frozen" and only the weights of the classifier got updated during training.

##### Step 1: Creating base model MobileNet V2

Base model used in this research was from the MobileNet V2 model were developed by Google developers with more than 1000 categories.

First, we picked intermediate layers of the MobileNet V2 model for feature extraction. A common practice is to use the output of the very last layer before the flatten operation, the so-called "bottleneck layer". The reasoning here was that the following fully-connected layers is too specialized, thus, the features learned by these layers are not really useful for a new task. The bottleneck features, however, retained much generality.

The following code used to initialize weight from the weight of MobileNet V2 model and define the dimension of the input to be 224x224 with 3 channels.

IMAGE\_SIZE = 224

IMG\_SHAPE = (IMAGE\_SIZE, IMAGE\_SIZE, 3)

# Create the base model from the pre-trained model MobileNet V2

base\_model = tf.keras.applications.MobileNetV2(input\_shape=IMG\_SHAPE,

include\_top=False,

weights='imagenet')

By specifying the include\_top=False argument, we loaded a network that does not include the classification layers at the top, which is ideal for feature extraction.

##### Step 2: Feature extraction

The convolutional base created from the previous step was used as a feature extractor. Then, we added a classifier on top of it and trained the top-level classifier.

base\_model.trainable = False

Add a classification head

model = tf.keras.Sequential([

base\_model,

tf.keras.layer.Conv2D(32, 3, activation='relu'),

tf.keras.layer.Dropout(0.2),

tf.keras.layer.GlobalAveragePooling2D(),

tf.keras.layer.Dense(8, activation='softmax')

])

##### Step 3: Model compilation

It was necessary to compile the model before training

model.compile(optimizer=tf.keras.optimizers.Adam(),

loss='categorical\_crossentropy',

metrics=['accuracy'])

##### Step 4: Model training

epochs = 10

history = model.fit(train\_generator,

epochs=epochs,

validation\_data=val\_generator)

#### Fine tuning

Fine tuning is re-training a transferred model (model trained by transfer learning method) to further improve performance and accuracy by reusing the top-level layers of the pre-trained model for new data set.

When performing the extraction before, we only trained a few classes on the MobileNet V2 base model. The weight of the pre-trained model did not change during training. Thus, in order to increase the model's performance, we trained (or "fine-tuned") the weights of the top layers of the pre-trained model alongside the training of the classifier you added. The training process forced the weights to be tuned from generic features maps to features associated specifically to our dataset.

##### Step 1: Un-freezing the top layers of the model

All we did here is unfreezing the base\_model and setting the bottom layers to be untrainable. Then, the model was recompiled, which was necessary for these changes to take effect, and resumed training.

base\_model.trainable = True

# Fine tune from this layer onwards

fine\_tune\_at = 100

# Freeze all the layer before the `fine\_tune\_at` layer

for layer in base\_model.layer[:fine\_tune\_at]:

layer.trainable = False

##### Step 2: Compiling model

model.compile(loss='categorical\_crossentropy',

optimizer = tf.keras.optimizers.Adam(1e-5),

metrics=['accuracy'])

##### Step 3: Training the model again

history\_fine = model.fit(train\_generator,

epochs=5,

validation\_data=val\_generator)

#### Converting model to TensorFlow Lite model

TensorFlow supports TensorFlow Lite library that allows us to operate with more consistent performance on Raspberry Pi.

TFLiteConverter was used to perform converting with *saved\_model\_dir* - the path containing the model we had already trained with convert() function - in serialized form (an array of bytes).:

tflite\_model = tf.lite.TFLiteConverter.from\_saved\_model(*saved\_model\_dir*).convert()

with open('model.tflite', 'wb') as f:

f.write(tflite\_model)

After that, we gained the model with compatible format for TensorFlow Lite, which was saved as ‘model.tflite’.

The following TensorFlow Lite interpreting processes are implemented based on the sample codes provided by MarcusLG users on Github[[18]](#footnote-18)

#### Implementing classification program

##### Step 1: Initializing TensorFlow Lite interpreter

‘.tflite’ model was loaded into memory, so the memory contained the graphs (graph) that can be executed (model’s execution graph). At the same time, input data was necessary to be formatted by get\_input\_details() function. Similarly, the format of the output data was also formatted with get\_output\_details() function.

interpreter = tf.lite.Interpreter(model\_path) #model\_path là đường dẫn đến file .tflite

interpreter.allocate\_tensors()

input\_details = interpreter.get\_input\_details()

output\_details = interpreter.get\_output\_details()

##### Step 2: Pre-processing of the input image

In this step, the image was converted into an array of a specified size (img\_row x img\_column) and a particular colour channel was extracted by using module TensorFlow.keras.preprocessing.image

import TensorFlow.keras.preprocessing.image as image

import numpy as np

img = image.load\_img(img[0], target\_size=(224, 224))

input\_img = image.img\_to\_array(img)

input\_img = np.expand\_dims(input\_img, axis=0)

input\_img = tf.keras.applications.mobilenet.preprocess\_input(input\_img)

x\_matrix = np.array(input\_img , dtype=np.float32)

x\_matrix = x\_matrix.reshape(input\_details[0]['shape'])

##### Step 3: Running the Interpreter

interpreter.set\_tensor(input\_details[0]['index'], x\_matrix)

interpreter.invoke()

##### Step 4: Extracting results

Results returned from the model in the form of Tensor, which was retrieved from the following method:

y\_matrix = interpreter.get\_tensor(output\_details[0]['index'])

The result should be converted to a one-dimensional array for easier identification by using numpy flatten() function:

feature = y\_matrix.flatten()

The feature variable was an array containing real numbers, which represented the percentage corresponding to the value that the classification model evaluates on each class (garbage categories). The sum of this array is 100%, with the highest-percentage result is the most feasible result after the prediction.

### Accuracy evaluation

###### Algorithm test

The accuracy of prediction result was performed by ImageDataGenerator class. It allowed us to divide training-used dataset into two parts including training and validation. We are based on the validation dataset to determine the accuracy in the model. The accuracy of the prediction reached 86%. The below figure (Figure 6) demonstrated the evaluated result:

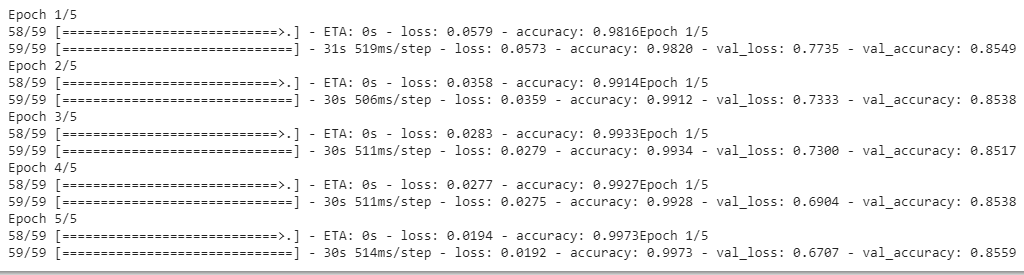


Figure 6. Algorithm testing result

Result in experimental test will be described later in the experimental results part.

#### Implementing classification program on Raspberry Pi

Because the Raspberry Pi could be considered as a mini computer, running a program on Raspberry Pi has been quite similar to running on a regular computer. First, we installed the necessary libraries for the program.

TensorFlow Library (TensorFlow Lite): Although the interpreter could be directly used from TensorFlow Lite library, the number of documentations for installing the Lite version on Raspberry was still limited, which caused difficulty for us during the implementation process. Therefore, we installed the entire TensorFlow library (which contains TensorFlow Lite library) for the device by Python's pip (or pip3) library manager as following

sudo apt install libatlas-base-dev

pip install TensorFlow

However, there were some errors when TensorFlow Lite library called:

undefined symbol: \_ZN6tflite12tensor\_utils39NeonMatrixBatchVectorMultiplyAccumulateEPKaiiS2\_PKfiPfi

It seemed that pip3 (and pip) had some problems and this issue had not been resolved yet, so we decided to install this library by using the whl file provided by user 'PINTO0309' on github[[19]](#footnote-19) (after unsuccessfully testing other options). This solution was not an official installation provided by TensorFlow, but we were unable to evaluate if it would cause any troubles.

numpy library: installed simply and successfully by the following command

pip install numpy

After all libraries have been successfully installed, the program could run normally

#### Connecting classification results with other supported devices

After the classification program has been successfully run on Raspberry, connecting the Raspberry Pi with other support devices could help the trash bin run automatically:

##### Raspberry Pi camera module

picamera and time libraries were used to manage the camera.

from picamera import PiCamera

from time import sleep

import wasteclassify as wcl

camera = PiCamera() #initials camera

camera.start\_preview() #gets the view of currently shooting stream of pictures

sleep(1) #previews in 1 seconds

camera.stop\_preview() #stop previewing

camera.capture('/home/pi/Downloads/test/image.jpg') #captures & saves picture into the path

path = '/home/pi/Downloads/test/image.jpg'

wcl.classifyWaste(path) #classifies image saved in ‘path’ variable

##### PIR sensor

RPi.GPIO library was imported to manage the pins.

import RPi.GPIO as IO

#imports other libraries and performs setups

#...........

PIR = 15

IO.setup(PIR, IO.IN) #receive signal from PIR sensor connected to pin 15

while 1:

sleep(1)

if(IO.input(PIR)==True): #there is a person/ animal moving in the surrounding area

#captures, analyzes & gives classified results as operation mentioned in camera section

#....

elif(IO.input(PIR)==False): #there is no moving signal in the surrounding area

print('sleeping...') #does nothing

##### Servo motors:

Two motors were responsible for rotating a temporary garbage container to the correct sub-bin to drop the garbage (2 sub-bins of 2 main categories). In details, the temporary container was placed in a standard position (fixed position) when classification was not required. That container consisted of a lower plate and an upper frame, each component was controlled by a servo motor.

Once the sub-bin number was determined, the upper frame rotated with an angle of 90 degree to push the garbage out of the temporary container. As a result, the garbage fell into the determined sub-bin and the upper frame returned back to their original position.

The detailed implementation codes are described as below:

#setup servo

servo1PIN = 17 #pin for servo to round

GPIO.setwarnings(False)

GPIO.setup(servo1PIN, GPIO.OUT)

p1 = GPIO.PWM(servo1PIN, 50) #p1 is the lower plate

p1.start(7.5) #sets standard position for the lower plate

#2 main categories:

box1 = ['PlasticBottle', 'Pen', 'Paper', 'Can']

box2 = ['StyrofoamContainer', 'PlasticBag', 'SweetPacket', 'PlasticGlass']

while 1:

sleep(1)

binNum = 0

if(GPIO.input(PIR)==True): #there is a person/ animal moving in the surrounding area

#captures, analyzes & gives classified results as operation mentioned in camera section

#....

#if the sensor operates, this part will classify

if(GPIO.input(pir\_sensor\_pin)==True):

sleep(2)

#rotates servo

if result in box1:

p1.ChangeDutyCycle(2.5)

time.sleep(2)

elif result in box2:

p1.ChangeDutyCycle(12.5)

time.sleep(2)

else:

break

p1.ChangeDutyCycle(7.5) #returns to the standard position

elif(GPIO.input(PIR)==False): #there is no moving signal in the surrounding area

print('Sleeping....') #does nothing

### Experimental results

Results are evaluated as 2 main parts

#### Classifying

The results of the classification were almost correct with some garbage samples such as cans, bottles, plastic cups, pens or styrofoam box (our subjective judgement during the experiment). However, confusion still rarely happened with plastic bags and food packs because these two types might have similar colors and shapes. This problem did not make huge concerns because these two types of garbage were classified in the same category (non-recyclable). Besides, paper waste samples sometimes gave inaccurate results (crumbled papers gave wrong result as 'plastic bags') because this category had too many different shapes that the current dataset was not able to cover. Moreover, the dataset was also lack of 'paper boxes' data, which led to 'foam boxes' result. In general, the practical testing process gave an accuracy of about 50-60% for classifying recyclable and non-recyclable trash. Therefore, it is essential to collect more data to increase the accuracy.

#### Dropping garbage

In general, dropping the garbage into the sub-bin worked well, which means the trash bin could put the classified garbage into the correct determined sub-bin. However, at the experimental level, the lower plate and the upper frame were quite small, it was only possible to classify small sized garbage. This problem can be solved by using more suitable materials and linking the details tightly, which helps the trash bin to be more stable to carry bigger temporary container.

## Mobile application

### Overview

This mobile application provides user with 3 main services:

Displaying: fetches information from database and shows on the screen for users to observe the information as well as the status (full or normal) of the trash bin

Managing: allows users to add trash bin to their working list for getting the information and modifying some description content based on their intention. Users can also remove the trash bin from their working list

Notifying: whenever some sub-bins of the trash-bin are full, user can get the notification so they can clean their trash bin if they have that trash bin on their working list

### Design

This application is built by using Flutter framework and Firebase platform, which provides Cloud Firestore database and other services such as Cloud Messaging and Cloud Function

#### Cloud Firestore database

Cloud Firestore[[20]](#footnote-20) is a “flexible, scalable database for mobile, web, and server development from Firebase and Google Cloud Platform.” It allows us to keep our data in sync across client apps through real-time listeners and offers offline support for mobile and web, which allow us to build responsive application that work regardless of network latency or Internet connectivity.

##### Some definitions in Cloud Firestore (Figure 7):

|  |  |
| --- | --- |
| - Collection: containers for your documents (similar to table in SQL model)  - Document: contains fields mapping to values (similar to column in SQL model)  - Field: a pair of key and value, supported many different [data types](https://firebase.google.com/docs/firestore/manage-data/data-types), from simple strings and numbers, to complex, nested objects such as sub-collection.(key is similar to row and value is the data stored in SQL model)  - Sub-collection: small collection inside documents | Figure 7. Firestore’s document storage  Source:<https://firebase.google.com/docs/firestore> |

#### Firebase Cloud Messaging

Firebase Cloud Messaging (FCM)[[21]](#footnote-21) is a cross-platform messaging solution that lets you reliably deliver messages at no cost. In this research, FCM was used to notify a client app whenever its tracking trash-bin is full.

#### Cloud Function for Firestore

Cloud Functions (<https://firebase.google.com/docs/functions/use-cases>) gives developers access to Firebase and Google Cloud events, along with scalable computing power to run code in response to those events. While it's expected that Firebase apps will use Cloud Functions in unique ways to meet their unique requirements, typical use cases might fall into these areas:

-  [Notify users when something interesting happens](https://firebase.google.com/docs/functions/use-cases#notify_users_when_something_interesting_happens).

-  [Perform Realtime Database sanitization and maintenance](https://firebase.google.com/docs/functions/use-cases#perform_sanitization_and_maintenance).

-  [Execute intensive tasks in the cloud instead of in your app](https://firebase.google.com/docs/functions/use-cases#execute_intensive_tasks_in_the_cloud_instead_of_in_your_app).

-  [Integrate with third-party services and APIs](https://firebase.google.com/docs/functions/use-cases#integrate_with_third-party_services_and_apis).

With Cloud Functions, we can deploy Node.js code to handle events triggered by changes in your Cloud Firestore database. This allows us to easily add server-side functionality into our app without running our own servers.

### Database design

#### Database description

This application is the initial version only for managing the trash bin information so the design is simple. There are two collections (Figure 8):

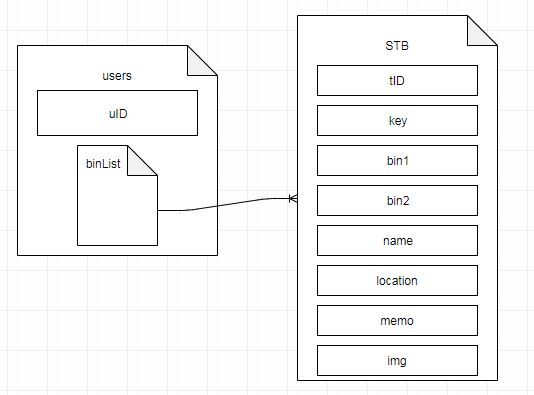


Figure 8. Collection model

#### Database Dictionary

Table 1. Database dictionary

|  |  |  |
| --- | --- | --- |
| **No.** | **Collection’s name** | **Description** |
| 1 | users | Stores users id |
| 1.1 | binList | A sub-collection belonging to collection ‘users’ to store the id of the trash bin which users has in their working list |
| 2 | STB | Stores id, sub-bins status and other information to manage the trash bin |

### Functions

#### Getting user ID function

* Working state: passive
* Trigger: when user open application
* Purpose: getting available id or generating new id for that user, which stored on both Cloud Firestore and local database helps the system now which user is using the application without requirement of login step.
* Constraint: Internet requirement
* Parameter required: none
* Method:

Table 2. Get user ID method

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Collection** | **Method** | | | |
| **Add** | **Edit** | **Delete** | **Query** |
| 1 | Users | X |  |  | X |
| 1.1 | binList |  |  |  |  |
| 2 | STB |  |  |  |  |

* Workflow

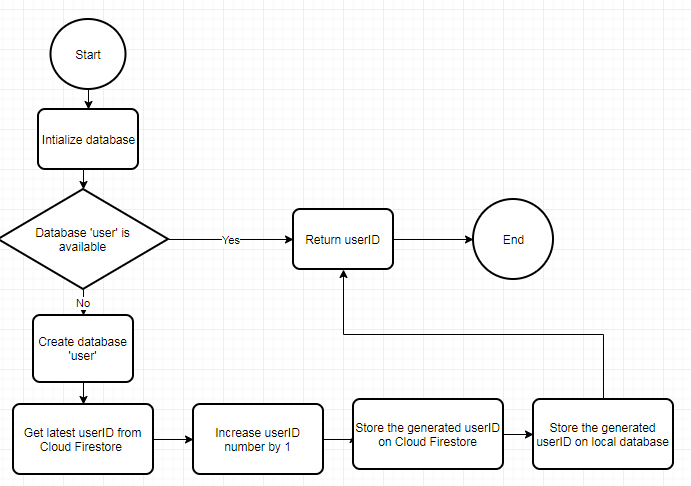


Figure 9. Get user ID function workflow

#### Listing function:

* Working state: passive
* Trigger: every time user opens the application
* Purpose: showing the list of the trash bin for further management options
* Constraint: none
* Parameter required:

Table 3. Listing function parameter

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Parameter name** | **Meaning** | **Example** |
| 1 | uID | User ID | U0002 |

* Method:

Table 4. Listing function method

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Collection** | **Method** | | | |
| **Add** | **Edit** | **Delete** | **Query** |
| 1 | Users |  |  |  | X |
| 1.1 | binList |  |  |  | X |
| 2 | STB |  |  |  | X |

* Workflow



Figure 10. Listing function workflow

#### Adding trash bin function

* Working state: active
* Trigger: when user tap on ‘+’ button on from the list screen.
* Purpose: adding a trash bin to user working list
* Constraint: none
* Parameter required:

Table 5. Adding trash bin function parameter

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Parameter name** | **Meaning** | **Example** |
| 1 | tID | Trash bin ID | TB010 |
| 2 | Key | Works as a ‘password’, which allow only user(s) who has/ have the key to access the information of that trash bin | ‘aE&7fz$1’ |

* Method:

Table 6. Adding trash bin function parameter

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Collection** | **Method** | | | |
| **Add** | **Edit** | **Delete** | **Query** |
| 1 | Users | X |  |  | X |
| 1.1 | binList | X |  |  |  |
| 2 | STB |  |  |  | X |

* Workflow

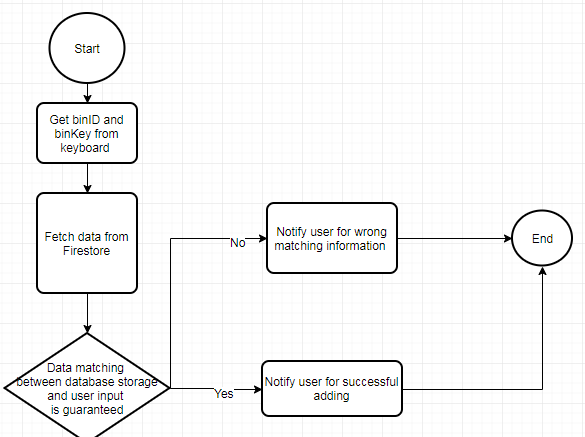


Figure 11. Adding trash bin function workflow

#### Showing information function:

* Working state: active
* Trigger: when user tap on a specific trash bin from the list screen.
* Purpose: showing the detailed description information of a specific trash bin
* Constraint: none
* Parameter required:

Table 7. Showing information function parameter

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Parameter name** | **Meaning** | **Example** |
| 1 | tID | Trash bin ID | TB010 |

* Method:

Table 8. Showing information function method

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Collection** | **Method** | | | |
| **Add** | **Edit** | **Delete** | **Query** |
| 1 | Users |  |  |  | X |
| 1.1 | binList |  |  |  | X |
| 2 | STB |  |  |  | X |

* Workflow



Figure 12. Showing information function parameter workflow

#### Adding and editing information functions:

* Working state: active
* Trigger: when user tap on corresponding icons.
* Purpose: adding new (when not created yet) or editing available detailed description information of a specific trash bin
* Constraint: none
* Parameter required:

Table 9. Adding and editing information functions parameter

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Parameter name** | **Meaning** | **Example** |
| 1 | tID | Trash bin ID | TB010 |

* Method:

Table 10. Adding and editing information functions method

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Collection** | **Method** | | | |
| **Add** | **Edit** | **Delete** | **Query** |
| 1 | Users |  |  |  |  |
| 1.1 | binList |  |  |  |  |
| 2 | STB |  | X |  | X |

* Workflow



Figure 13. Adding and editing information functions workflow

#### Removing trash bin function:

* Working state: active
* Trigger: called when user tap on ‘x’ icon of a specific trash bin.
* Purpose: deleting a specific trash bin from user’s working list
* Constraint: none
* Parameter required:

Table 11. Removing trash bin function parameter

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Parameter name** | **Meaning** | **Example** |
| 1 | tID | Trash bin ID | TB010 |

* Method:

Table 12. Removing trash bin function method

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Collection** | **Method** | | | |
| **Add** | **Edit** | **Delete** | **Query** |
| 1 | Users |  |  | X |  |
| 1.1 | binList |  |  | X |  |
| 2 | STB |  |  |  |  |

* Workflow

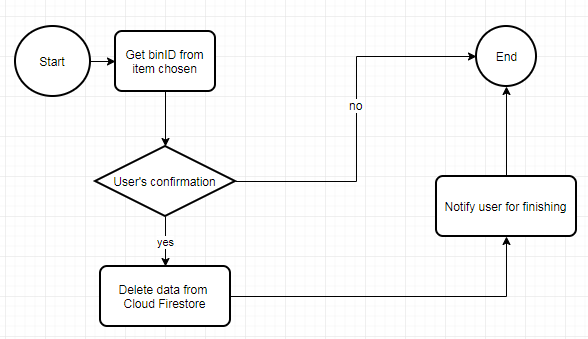


Figure 14. Removing trash bin function workflow

#### Notifying function:

This function is developed not by using Flutter but by Cloud Function to trigger the Cloud Messaging service.

* Working state: passive
* Trigger: when sub-bin(s) of a trash bin become(s) full.
* Purpose: notifying users there is (are) some full sub-bin so they can know and clean the sub-bin(s).
* Constraint: Internet requirement
* Parameter required: none
* Method:

Table 13. Notifying function method

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Collection** | **Method** | | | |
| **Add** | **Edit** | **Delete** | **Query** |
| 1 | Users |  |  |  |  |
| 1.1 | binList |  |  |  |  |
| 2 | STB |  |  |  | X |

* Workflow

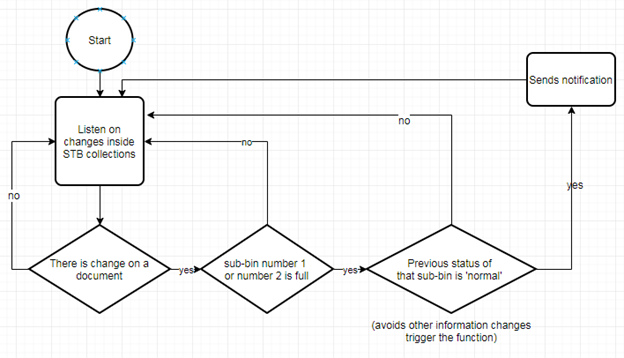


Figure 15. Notifying function workflow

## Whole system

### System Architecture

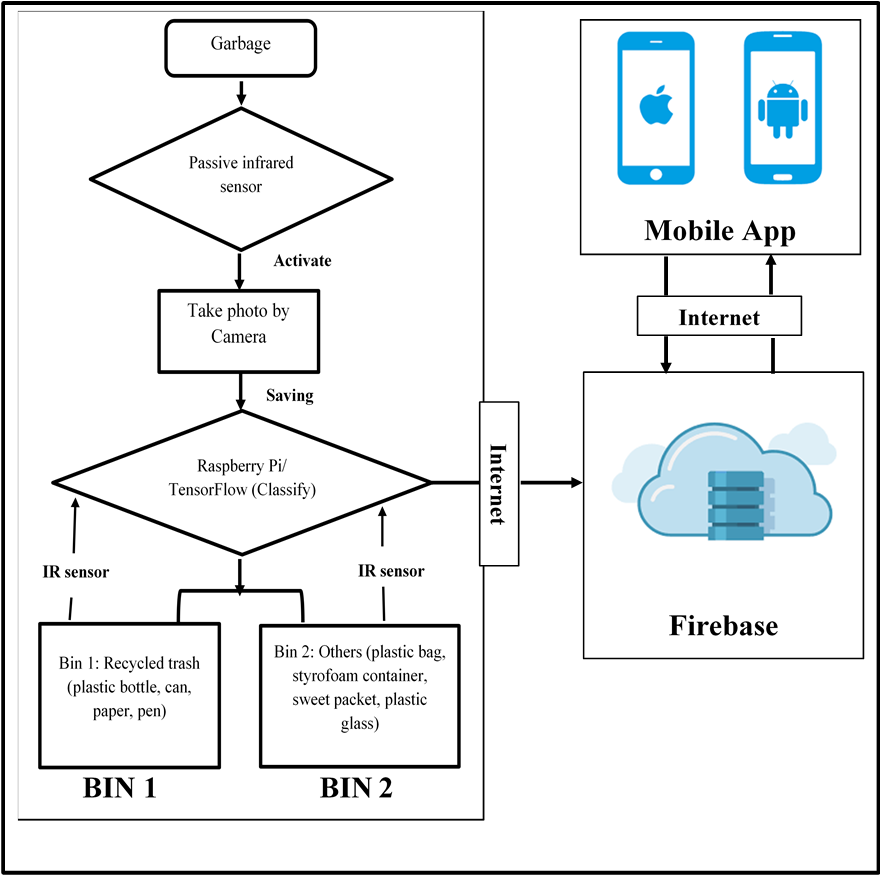
****

Figure 16. System architecture

The trash bin works normally as in the trash bin model section. After connecting the whole system, whenever the trash is full, the signal is sent to Cloud Firestore database. This data is transferred to the mobile application in real-time mode.

### Status detection

We installed an IR sensor near the top of every sub-bin. The sensor, which returned a signal when it detected any object in its range, allowed us to know if the trash inside reach its height, which meant that the sub-bin bin was full

### Connection building

The connection between the trash bin and the mobile application was built simply using Cloud Firestore Database as an ‘intermediate bridge’. The status of the trash bin (full or not) was uploaded on Cloud Firestore and with the help of real-time techniques this database offered, the application can receive the notification immediately as long as Internet connection was available.

In order to do that, the program written in Python should include the code for Cloud Firestore initialization:

cred = credentials.Certificate("/home/pi/Downloads/smart-tb-firebase-adminsdk-1ofks-e1dc5f4b8f.json") #.json file with the credentials needed to access the Firebase project

#only project owner can generate credentials file from Firebase Service Account Console

# initialize cloud firestore

firebase\_admin.initialize\_app(cred, {

'projectId': 'smart-tb', #our project id on Firebase platform

})

db = firestore.client()

### Data Updating

Since Firestore was a commercial platform (it still offered free quota) and its billing was based on read, write and delete counting[[22]](#footnote-22) we had to be careful of uploading information to Firestore to prevent huge write counting, which could lead to free quota limitation exceeding. Therefore, tracking the trash bin status and updating data per second in while loop was not our approach. The solution used was that the status tracking progress still run per second but we only wanted to update the data whenever a change happened, for example, from not-full to full or vice versa. Below is the code for tracking the first sub-bin (recyclable trash) and updating its status to Cloud Firestore database:

#bin1

if (is\_full1==False and GPIO.input(box\_1\_pin)==False):

#bin1 change state from not-full to full

doc\_ref = db.collection(u'STB').document(u'T0001')

doc\_ref.update({

u'bin1': True,

})

is\_full1 = True

if (is\_full1==True and GPIO.input(box\_1\_pin)==True):

doc\_ref = db.collection(u'STB').document(u'T0001')

doc\_ref.update({

u'bin1': False,

})

is\_full1 = False

# SUMMARY

### Summary

In conclusion, we successfully developed an automatically classifying trash bin with good results. We also designed an experimental model to ensure that the garbage could be dropped into the correct sub-bin after the classification process.

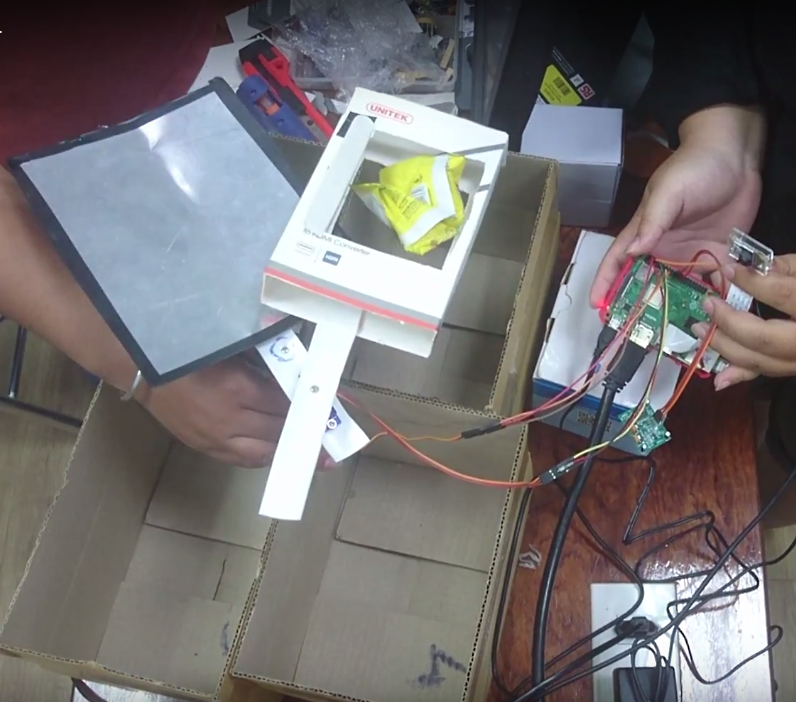


Figure 17. Experiment picture

However, there have been some existing issues:

* Materials used to build the trash bin were quite simple, which leads to a situation that the product created can be only used for testing purposes but not for a fully finalized deployment.
* The dataset used was not diverse enough, which led to wrong classification in some cases

### Development orientation

#### Short-term orientation

With the achieved results and the shortcomings as mentioned above, we want to improve the following factors:

* Adding a screen to display classifying results for observing classification results better
* Collecting more data for the dataset: collect appropriate images to improve the accuracy for the classifying result
* Improving the design: modifying the current design to increase the number of the sub-bins so that the trash bin can work with more types of waste such as hazardous waste or technical waste.
* Reducing the production cost: using equivalent modules and devices with lower price to reduce the production costs.
* Optimizing the productivity: optimizing classifying time to meet requirements if the product is deployed at public places.

#### Long-time orientation

We want to deploy the product after finishing the development and installation for trial usage in some areas. In order to do that, we will monitor the working process of the modules and devices used (Raspberry Pi, camera, servo motor, PIR SR501 sensor) to perform modifying and improving the design to meet user requirements. In the future, we hope that this product can be widely used in more families to ensure that people, especially children, are able to classify waste easily from their own homes. In addition, we will continue our research to develop a smart robot model that can pick up trash automatically and classify the trash by itself.

# REFERENCES

|  |  |
| --- | --- |
| [1] | R. A. P. S. P. R. &. G. R. Sarda G., "SMART TRASH CAN USING IOT.," April 2019. |
| [2] | Chandramohan, A., Mendonca, J., Shankar, N. R., Baheti, N. U., Krishnan, N. K., & Suma, M. S. ). Automated waste segregator. In *2014 Texas Instruments India Educators' Conference* 2014, April. |
| [3] | M. a. T. G. Yang, "Classification of trash for recyclability status," *CS229 Project Report,* vol. 2016, 2016. |
| [4] | Costa, B., Bernardes, A., Pereira, J., Zampa, V., Soares, E., Soares, C., & Silva, A. “Artiﬁcial Intelligence in Automated Sorting in Trash Recycling”. 2018. |
| [5] | B. A. P. J. Z. V. S. E. S. C. &. S. A. Costa B., "Artificial Intelligence in Automated Sorting in Trash Recycling," in *Anais do XV Encontro Nacional de Inteligência Artificial e Computacional*, 2018. |
| [6] | V. H. Tiệp, "Bài 2: Phân nhóm các thuật toán Machine Learning.," 27 December 2016. [Online]. Available: https://machinelearningcoban.com/2016/12/27/categories/. [Accessed 15 August 2019]. |
| [7] | T. Đ. Tâm, "Bắt đầu với Machine Learning thông qua TensorFlow (Phần I.2).," 17 July 2017. [Online]. Available: https://kipalog.com/posts/Bat-dau-voi-Machine-Learning-thong-qua-TensorFlow--Phan-I-2. [Accessed 15 August 2019]. |

1. [Bishop, C. M.](https://en.wikipedia.org/wiki/Christopher_M._Bishop) (2006), *Pattern Recognition and Machine Learning*, Springer, [ISBN](https://en.wikipedia.org/wiki/International_Standard_Book_Number) [978-0-387-31073-2](https://en.wikipedia.org/wiki/Special:BookSources/978-0-387-31073-2) [↑](#footnote-ref-1)
2. Muslim, Jameel Syed, et al. "Automatic Image Annotation for Small and Ad hoc Intelligent Applications using Raspberry Pi." *MATEC Web of Conferences*. Vol. 255. EDP Sciences, 2019. [↑](#footnote-ref-2)
3. ACKLEY, H. Sprague. *Optical character recognition systems and methods*. U.S. Patent Application No 15/793,407, 2019. [↑](#footnote-ref-3)
4. Borenstein, Jason, Joseph R. Herkert, and Keith W. Miller. "Self-driving cars and engineering ethics: the need for a system level analysis." *Science and engineering ethics* 25.2 (2019): 383-398. [↑](#footnote-ref-4)
5. TensorFlow: open source machine learning from Google studio in Youtube posted on 09.11.2015. For more information on obtaining material refer to https://www.youtube.com/watch?v=oZikw5k\_2FM [↑](#footnote-ref-5)
6. TensorFlow Lite Guide from TensorFlow.org. For more information on obtaining material refer to https://www.tensorflow.org/lite/guide [↑](#footnote-ref-6)
7. Raspberry Pi 3 Model B+ (Made in UK) from Hshop.vn . For more information on obtaining material refer to https://hshop.vn/products/combo-raspberry-pi-3-model-b-special [↑](#footnote-ref-7)
8. Camera Raspberry Pi V1 5MP from Hshop.vn For more information on obtaining material refer to https://hshop.vn/products/camera-raspberry-pi [↑](#footnote-ref-8)
9. RC Servo MG90S from Hshop.vn For more information on obtaining material refer to https://hshop.vn/products/dong-co-rc-servo-mg90s [↑](#footnote-ref-9)
10. Mdl50 Module Motion Sensor Hc-Sr501 from robocon.vn For more information on obtaining material refer to http://robocon.vn/detail/mdl50-module-cam-bien-chuyen-dong-hc-sr501.html [↑](#footnote-ref-10)
11. Video introduction about Fluter in fluter.dev. For more information on obtaining material refer to https://flutter.dev/docs/resources/videos [↑](#footnote-ref-11)
12. Park Byung-hwa. Google's web programming language Dart, why learn? (01.04.2019). These documents can be researched at codingworldnews.com [↑](#footnote-ref-12)
13. [Wm Leler](https://hackernoon.com/@wmleler1)**.** Why Flutter uses Darts (22.12.2017). For more information on obtaining material refer to https://hackernoon.com/why-flutter-uses-dart-dd635a054ebf [↑](#footnote-ref-13)
14. Material design guideline last updated on 25.07.2019 in material.io. For more information on obtaining material refer to https://material.io/design/ [↑](#footnote-ref-14)
15. Welcome to Firebase! In console.firebase.google.com. For more information on obtaining material refer to https://firebase.google.com/?gclid=CjwKCAjwk93rBRBLEiwAcMapUdMa1m8iAumqPUt4Bvnljz3DxHY-fi\_IWrhGokpSJMLmI4G1\_o1LGBoCYSsQAvD\_BwE [↑](#footnote-ref-15)
16. What Is NoSQL. These documents can be researched in mongodb.com. For more information on obtaining material refer to https://www.mongodb.com/nosql-inline [↑](#footnote-ref-16)
17. Katariya, Y., & Garg, N. (2019). TensorFlow/examples. For more information on obtaining material refer to github.com/TensorFlow/examples/blob/master/community/en/flowers\_tf\_lite.ipynb [↑](#footnote-ref-17)
18. Lim, M. (2018). MarcusLG/TFlite-on-Python. Retrieved from [github.com/MarcusLG/TFlite-on-Python](https://github.com/MarcusLG/TFlite-on-Python). For more information on obtaining material refer to https://github.com/MarcusLG/TFlite-on-Python [↑](#footnote-ref-18)
19. TensorFlow-bin report PINTO0309. For more information on obtaining material refer to [https://github.com/PINTO0309/TensorFlow-bin](https://github.com/PINTO0309/Tensorflow-bin) [↑](#footnote-ref-19)
20. Cloud Firestore in firebase.google.com/docs/. For more information on obtaining material refer to https://firebase.google.com/docs/firestore [↑](#footnote-ref-20)
21. Firebase Cloud Messaging in firebase.google.com/docs/. For more information on obtaining material refer to https://firebase.google.com/docs/cloud-messaging [↑](#footnote-ref-21)
22. Usage and limits in [firebase.google.com/docs/firestore/quotas](https://firebase.google.com/docs/firestore/quotas) visited on 05.08.2019 [↑](#footnote-ref-22)