Boston University
Electrical & Computer Engineering
EC464 Senior Design Project

User Manual

Ecobin

by

Team 9
Ecobin

Team Members

Aditya Wikara adwikara@bu.edu
Hayato Nakamura hayaton@bu.edu
Kevin Sinaga kfsinaga@bu.edu
Shreenidhi Jayaram shreej@bu.edu
Charles Thao cthao19@bu.edu

User Manual

Table of Contents

Table of Contents 2

Executive Summary 3

Introduction 4

System Overview and Installation 5

2.1. Overview block diagram 5

2.2. User interface. 6

2.3. Physical description 6

2.4. Installation, setup, and support 8

Operation of the Project 10

3.1. Operating Mode 1: Normal Operation 10

3.2. Operating Mode 2: Abnormal Operations 11

3.3. Safety Issues 11

Technical Background 12

4.1. Hardware Component 12

4.2. Mechanical Component 12

4.3. Software Component 13

Cost Breakdown 16

Appendices 19

Appendix A - Specifications 19

Appendix B – Team Information 20

Executive Summary

Ecobin
Team 09

Ecobin is a smart trash can that automates recycling, making it an easier task in a modern home and therefore aiming to reduce garbage disposals to landfill and hence carbon footprint per person in America. All the user has to do is throw trash into the device, and Ecobin will automatically sort it into its appropriate compartment: recyclable or non-recyclable. The logic of Ecobin lies in the embedded system powered by a Raspberry Pi and several sensors. Its software functionality relies on computer vision, image recognition and cloud computing to determine the recyclability of an object. Ecobin also comes with an iOS application with an user friendly interface. In short, it is an up-and-coming Internet-of-Things device that would change and expedite recycling in an American household

1. Introduction

Since 1970s, many scientists have proved and studied the effects of industrial and urban activities on the environment. Climate change has caused from torrential flooding in Asia to monstrous wildfires in California, drove 50% of Earth's marine life to extinction, and imposed many public health crises around the globe. In the United States, while 80% of materials are recyclable, some 55% of materials end up in landfills; much of which is plastic wastes that can take up to 450 years to decompose. Within the city of Boston, a city with its progressive environmental agenda, only 21% of disposed materials are recycled in 2017.

As many studies and surveys suggest, many people lack the knowledge of how and what to recycle or hesitate due to the hassles of separating their wastes into different compartments. As a result, they put all items one trash bin. As Mr. Cootner, Ecobin's primary customer, expressed, he and many households believes in the importance of recycling, but the inconvenience or lack of knowledge often hinder them from doing so.

Ecobin wants to eliminate this dilemma by automating trash classification, leaving the user taking no extra steps to recycle other than just throwing their items away as usual. As an Internet-of-Things device, Ecobin classifies and sorts everyday trash as either "recyclable" or "trash". This project, aided by machine learning and embedded system, encourages a more sustainable lifestyle in order to curb the threats and effects of climate change in the light of environmentally taxing landfills. Ecobin combines a set of modern technologies for its functionality: microprocessing, computer vision and cloud computing.

The Ecobin is equipped with an iOS application that conveys classification results to the user for an easy interface to keep track of their recycling activities. The user can look at how much of their objects were recyclable and the accuracy of that prediction so that users can see to make sure that Ecobin correctly sorted it. Ecobin is an integrated product with machine learning, database and iOS integrated capabilities. Additionally, Ecobin also makes with error correcting capabilities in check. For example, when an item is thrown in and Ecobin isn't sure if it is a recyclable or a trash item, it automatically classifies the item as Trash and reports the doubted accuracy to the user. Such a precaution is kept in place to ensure that no items which is trash is mistakenly classified as recyclable in order to avoid false positive, when a landfill item ends up in the recycling compartment.

The Raspberry Pi (henceforth Raspi or Pi) acts as a control hub and which handles signal processing for a myriad of sensor as well as captures images capturing. On the cloud, the software backend of Ecobin categorizes an object placed in the bin as recyclable or trash. This setup renders a smooth user experience. With IoT perks coupled with robust motors, Ecobin serves as a replacement for the traditional trash can and hopes to reduce carbon footprint of its customers. Not only can it serve problems at home, Ecobin could be installed in public space such as office and malls.

2. System Overview and Installation

Ecobin consists of a semicircular platform for object detection, as well as a housing (box) to keep the environment enclosed. The semicircle is the platform on which the object is placed. The camera, RasPi, and LED strip are mounted on the left side of this platform. The motion sensor is positioned on top of the housing, right beside the lid. The sorting mechanism employed is a sweeper comprising the rear wall of the imaging stage, which turns clockwise, as seen from the top, if the object is general waste, and counterclockwise if recyclable. A stepper motor triggers this sweeping motion.

Figure 2.1 shows how the product's modules interact with each other. The input involves the trash object and the hand motion, which triggers the object detection process. In this module, the Ecobin takes a picture of the object, encodes it, and puts it in the API, which is in the cloud. The cloud backend will then classify the object as "Trash" or "Recyclable" using a machine learning algorithm. The backend then updates this information in the API, and stores it permanently in the MongoDB Atlas database. This process in the cloud roughly takes 1 second. Afterwards, the RasPi gets the classification of the object from the API, and signals the stepper motors. The output will involve the stepper motor moving clockwise for "Trash", and anti-clockwise for "Recyclable". Last of all, while these processes are happening, the Ecobin has two external functionalities: the iOS application and the capacity build up detection.

2.1. Overview block diagram

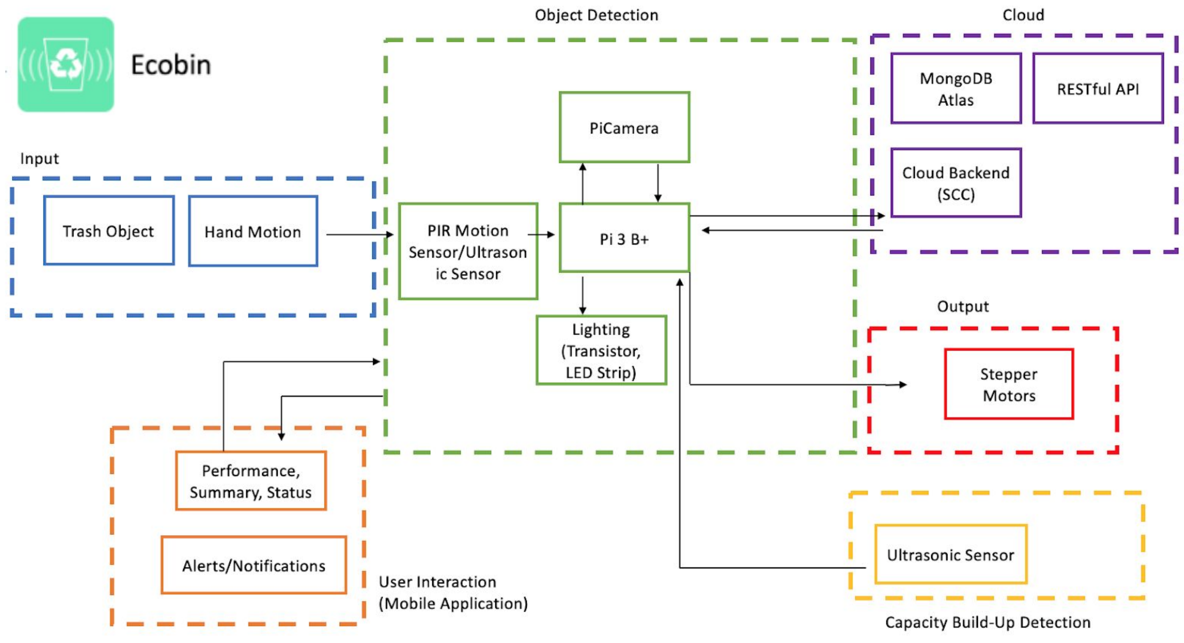
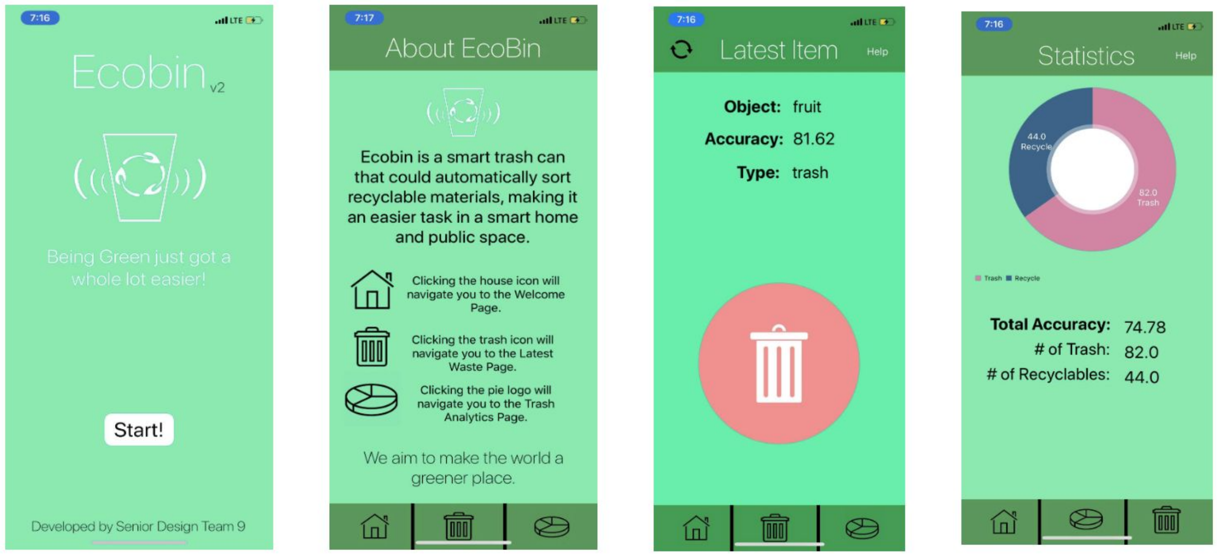


Figure 2.1: System Overview of the Ecobin.

2.2. *User interface.*

The user interface includes the bin itself, which will be activated by triggering the motion sensor. If the lid does not automatically lift, please hover your hands over the sensor. Any malfunctions should be addressed by contacting the developers.

The team also built an iOS application where the user than monitor and observe status, capacity as well as their performance. The UI will also update the status automatically when a new item is placed and sorted.



Ecobin is a smart trash can that could automatically sort recyclable materials, making it an easier task in a smart home and public space.

Clicking the house icon will navigate you to the Welcome Page.

Clicking the trash icon will navigate you to the Latest Waste Page.

Clicking the pie logo will navigate you to the Trash Analytics Page.

We aim to make the world a greener place.

Object:	fruit
Accuracy:	81.62
Type:	trash

Total Accuracy:	74.78
# of Trash:	82.0
# of Recyclables:	44.0

Figure 2.2: iOS Application UI

2.3. Physical description

Figures 2.3, 2.4, and 2.5 shows the main physical components of the Ecobin. Figure 2.3 is the upper part of the Ecobin, which is crucial for the classification of an object. Most of the hardware will be mounted on the sides of the semicircle. Figure 2.4 is the Ecobin's lid. The lid will have the PIR motion sensor attached to detect motion, which triggers the opening of the automated lid. Lastly, Figure 2.5 shows the overall structure of the Ecobin. The bottom part is essentially just two different receptacles for either "trash" or "recyclable" objects.

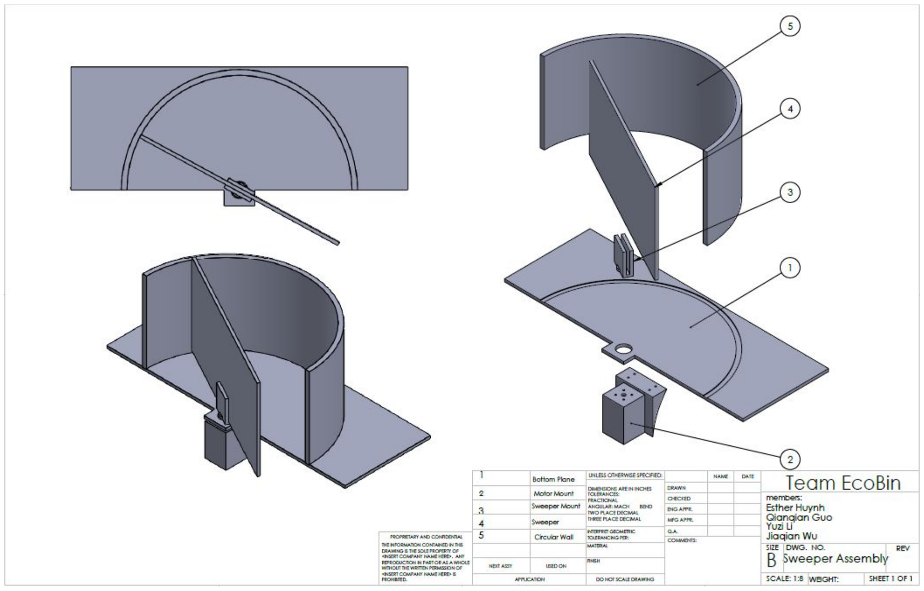


Figure 2.3: Sorting Mechanism (Sweeper)

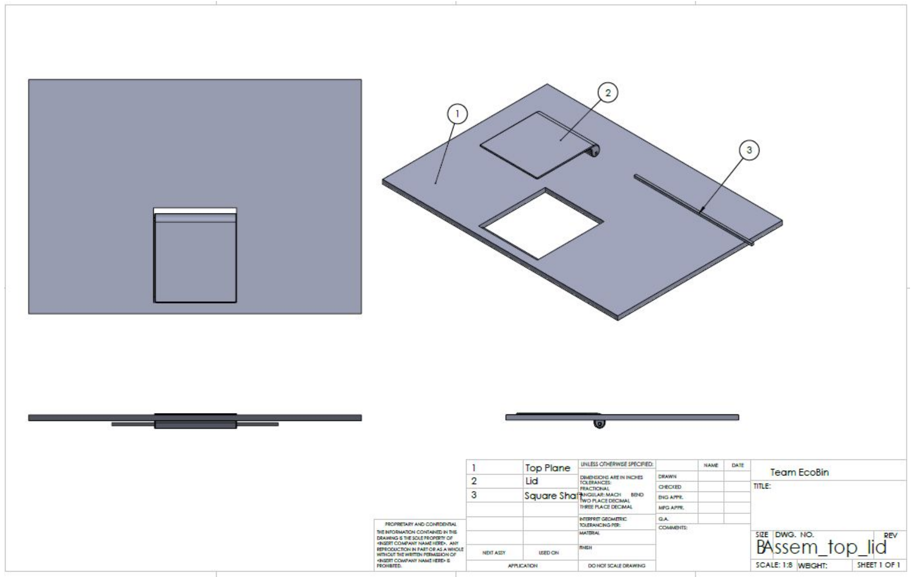


Figure 2.4: Automatic Lid

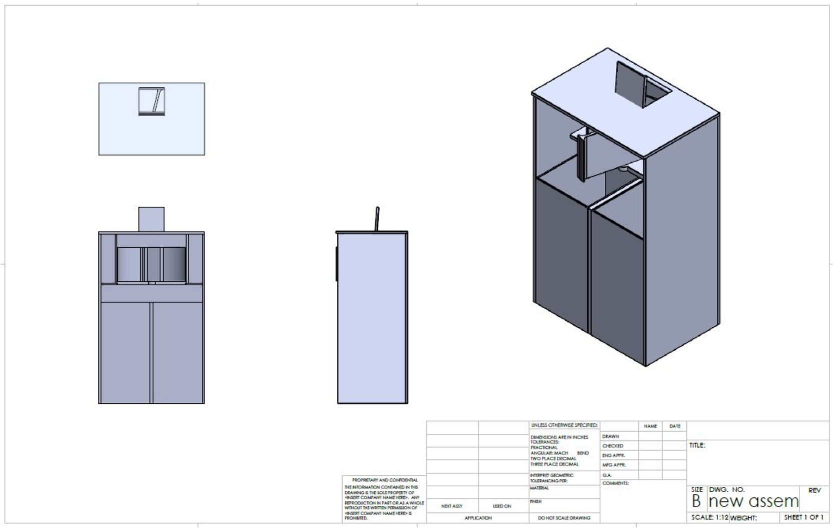


Figure 2.5: Overall Ecobin Structure

2.4. Installation, setup, and support

Installation: Each Ecobin device will come prebuilt with no further assembly required, except for the need to place the lid on the main body of the trash can. The user only needs to plug in the power supply cord and the system will be ready.

To maximize the functionality of the Ecobin, the user should consider several factors. The Ecobin is an IoT device that requires Internet connectivity. Therefore, it would be ideal to have the Ecobin positioned at a location that is directly exposed to the Wi-Fi router (minimal walls or doors from the router). Second, the Ecobin is a relatively large device for a trash can. The user needs to ensure that the installation facility has enough space for the Ecobin, and that there is easy access to electricity. Unlike conventional trash cans, the Ecobin is a smart trash can with a multitude of sensors. This means that the Ecobin should not be placed near sources of water. For example, if the user wants Ecobin in the kitchen, it should not be right next to the sink in case of any leakage or other emergencies.

Setup:

1. Download Ecobin iOS application from the App Store
2. Create user profile by setting up a login and password
3. Register the bin on the application. Each Ecobin device will come with a unique identification code.
4. Throw item into the Ecobin.
5. See results of classification and accuracy on the app.

Support: The user might find Ecobin intimidating due to the sensors and motors that are inside the device. This is why we provide a user interactive iOS mobile application. This mobile application allows the user to see whether the object is trash or recyclable, and the confidence level. This way, the user does not need to manually check the receptacles to ensure that the object is sorted in the correct compartment. In addition, all the information from the Ecobin will be stored in a cloud database. The mobile application will present this information in several forms such as recommended trash to recycling ratio, recycling rates, and type of product that is thrown most frequently. Last of all, the iOS app can also alert the user when the receptacles are at full capacity. This prevents the user from having to manually check the bin, hence saving the user a lot of time.

3. Operation of the Project

3.1. Operating Mode 1: Normal Operation

Activation

1. When the user wants to throw away an object, just approach the bin and it should be activated by the motion sensor.

   Notice: the current version only supports a single item at once. Multiple items could cause misclassification

2. The lid will automatically open, allowing the user to drop the object to the sorting platform.
3. After around 5 seconds, the lid will automatically close. If the lid opens even without motion, this may be an issue with the internal connections. If this happens, unplug the power supply and contact the Ecobin team.

Classification

1. After an object is disposed, the light will lit up. Ecobin will take a photo of an image and process it.
2. If the type is trash, the RasPi commands the sweeper to rotate clockwise. Conversely, if the type is recyclable, the sweeper will rotate anticlockwise.

   If the sweeper does not fully rotate, or if it makes sounds, immediately unplug the power supply. This may indicate that there is a connection issue, a short circuit, or either too much or not enough power driven into the motor controller.

Capacity Build Up Detection

1. The Ecobin also has a capacity build up detection functionality, enabled by ultrasonic sensors. The user could view this information in the iOS mobile application.
2. In the case where one or both compartments of the trash bag are close to full capacity, the iOS Application will alert the user about the need to replace trash bags, and will specify which compartment is full. The warning sign, which displays, "WARNING: *Recycle* compartment full" or "WARNING: *General Waste* compartment full", will continue to stay on the main page until the user has successfully replaced the indicated trash bag.

If the user is notified that full capacity has been reached even though the receptacle is not full, restart the system to recalibrate.

iOS Application

In addition to reading capacity, Ecobin has an iOS mobile application which provides an interface to interact with the user.

1. The application enables the user to check the status of the Ecobin as well as a variety of other useful data, such as local trash day information, recommended trash to recycling ratio, and the type of product that the user most frequently throws away.
2. From the display page, the application will show the latest trash data, more specifically if it was recyclable or trash, what the object exactly is, and the object detection confidence percentage level.
3. Additionally by clicking on the pie-chart logo on the bottom bar, the user will be able to see their total recyclable trash vs total general waste statistics along with the average overall confidence level value. There is also an interactive pie chart which helps the user visualise the information presented.
4. The user can tap the refresh button at the top left corner of the page to automatically fetch information if any new object is thrown into the Ecobin.

3.2. Operating Mode 2: Abnormal Operations

Hardware errors

1. If Ecobin fails to activates its lid, there could a processing error with the Raspberry Pi has occurred. Simply disconnect and reconnect the power.
2. If the motion sensor detects error, an interruption will be flagged. The Pi will automatically restart the script.

Classification

1. In Ecobin's default setting, when the confidence level falls below 70%, the waste object will automatically be sorted into the *trash* compartment.

   Note: this could be caused by an object not yet trained on the model.
2. If an object is misclassified, the user has an option to inform the team that it was misclassified and specify its proper class. This will improve user experience overtime.

   Note: only do this when the user is absolutely sure, as improper data might contaminate the dataset and influence the system's future performance.
3. If the system is connected to Wifi, it will display "Error. Please reconnect" on the iOS application.

3.3. Safety Issues

1. The Ecobin device does not have the capacity to handle power surges and so far has only been tested to operate on 110VAC at 60 Hz, or North American power standards, and might not work as well with other countries' power ratings. Users are advised to make sure that the Ecobin device is connected to an outlet with surge protection. As of now, it

is not advised to use the Ecobin device outside of North America until it is tested or certified in other countries.

2. Users should also be careful to not spill liquids. Spilling liquids on any other part of the Ecobin device may damage the exposed electronics, such as the motion sensor or ultrasonic sensors. Users are advised to dispose of any liquid within an object before throwing said object into the Ecobin.
3. Users should also take care when disposing objects, so as not to not take longer than 5 seconds to dispose of an object and to ensure that the object completely fits within the sorting cylinder of the device. The lid is only set to be open for 5 seconds, so the object must be completely within the cylinder at the end of the 5 seconds. If either the object is too large or takes longer than 5 seconds to get into the sorting cylinder, there is a risk that the motors would get jammed.

4. Technical Background

4.1. Hardware Component

The Ecobin uses a RasPi and several sensors to fulfill its hardware functionalities. A RasPi was selected because it essentially functions as a general purpose computer with Linux operating system. Python was used as the main programming language throughout the project, to control sensors and motors. Object detection was made possible with a Raspberry Pi camera, coupled with a wide angle lens to maximize image captured. It also utilized a 12V LED strip for proper illumination since the enclosure is dark in regular room setting. The LED strip is powered by a seperate power supply since the Pi can only supply up to 5V.

PIR motion sensor is used to detect hand motion when a user is trying to throw away an object into the Ecobin. The motion triggers an automatic lid and the LED strip. Initially, there were considerations of using an ultrasonic sensor to detect motion. However, the ultrasonic sensor would not work since the sound waves would deflect of that the casing. The ultrasonic sensor will act as a bin capacity detector and notify the user when the bin is full.

4.2. Mechanical Component

- A regular stepper motor (NEMA-17) was selected for the automatic lid because of its low torque requirements. Opening the lid requires ¼ of a full rotation. The key factor was to ensure the pulley teeth did not slip with the timing belt using a tensioner.
- The sweeper required a stronger motor, hence a stepper motor with a gearbox. This motor has a larger controller and is powered by a 12V power supply.
- The sorting mechanism design is based on a semicircle shaped platform. This enabled the need of only one mechanical actuator, which rotates clockwise to push the object into the "trash" receptacle, or anticlockwise for "recyclable".

4.3. Software Component

Ecobin operates with the cooperation of three main software components: frontend, API and backend. The frontend of the product includes an iOS application, developed with Swift 4 and XCode, while the API and backend were written in Python and runs on an OpenStack cloud.

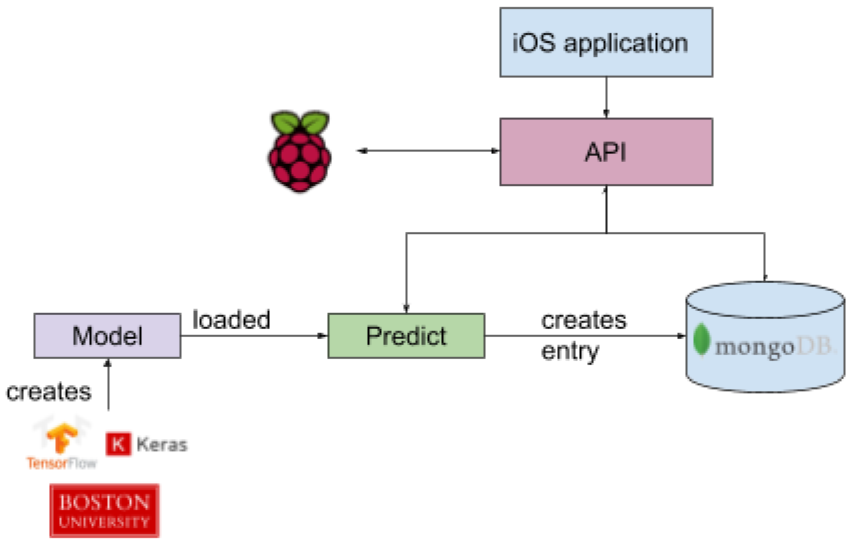


Figure 3.1: Overall Software Architecture

a. Object recognition
- Library used: Keras
- Version: 2.2.4
- Technique: transfer learning with fine tuning
- This deep learning neural network was trained to recognize 13 classes, with approximately 1000 images for training and validation.
- Training took place on Shared Computing Cluster at Boston University
- Specifications of cloud instance used for training:
    - 2 Nvidia P100 Graphical Processing Units (GPUs), with 16GB of memory each.
    - 128GB memory
    - Trained the model over 25 epochs with SGD optimizer, learning rate decay and Nesterov for better results.
    - Training accuracy (95%) and validation accuracy (94%)

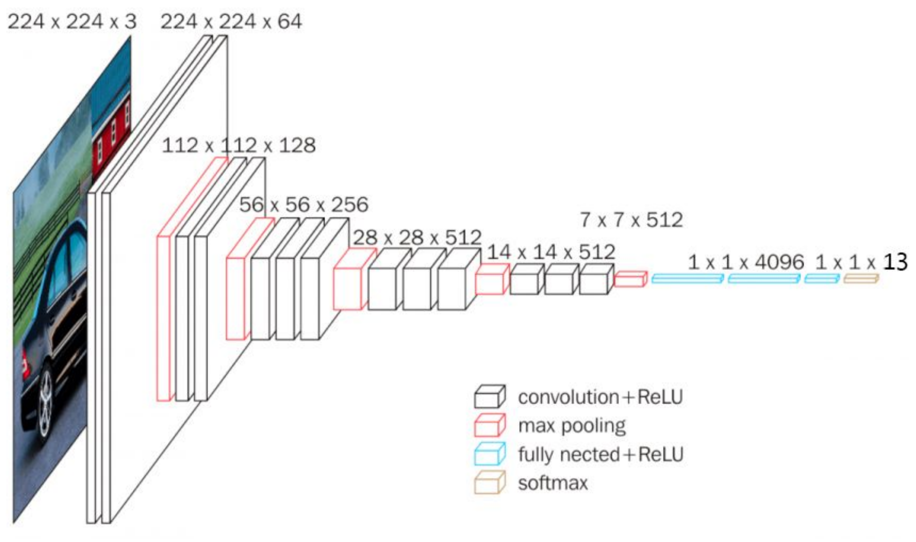


Figure 3.2: Visual Representation of the Keras Model

Different models were taken into considerations while developing the project:

Approach	Advantages	Disadvantages
Build model from scratch on PyTorch	● Specialized ● Self-governance on software	● Needs millions of images for training and validation ● Labor intensive ● Costly for cloud instances with GPUs ● Time consuming: training and testing could take hours before optimization
Google Net	● High accuracy	● Only 4 million parameters
VGG-16	● Pre-built and tested ● Open-source	● Redundant: contains 1000 classes ● Low accuracy
Transfer learning from VGG-15	● Robust feature extracting ● Hardware resource ● Time consuming: training could take hours	● Needs GPUs

The code and instruction for this data lies While gathering data for the project, images are stored in “train” and “validation” directories. Subsequently, the developer can modify and train the model using “transfer_learn.py” in the terminal, given sufficient computing powers are present.

b. API and MongoDB

Ecobin API is the primary tool for Raspberry Pi, iOS application and the backend to communicate. A architectural diagram of the API is as follows:

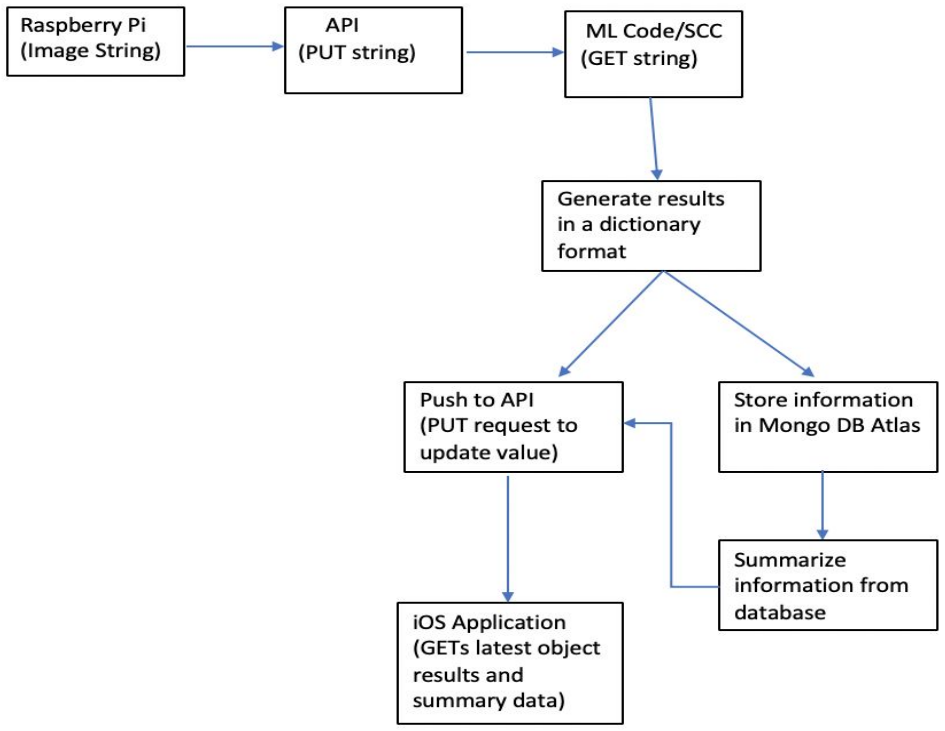
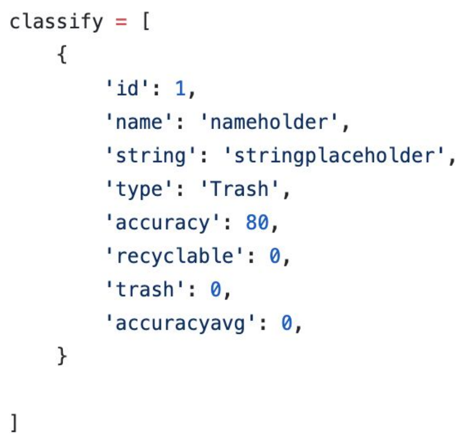

Raspberry Pi (Image String)
API (PUT string)
ML Code/SCC (GET string)
Generate results in a dictionary format
Push to API (PUT request to update value)
Store information in Mongo DB Atlas
Summarize information from database
iOS Application (GETs latest object results and summary data)

Figure 3.3: Software Flow Diagram

The technical principles and tools of this implementation is as follows:

1. The API was built using Flask and Python in integration with the Raspberry Pi, MongoDB database and iOS application.
2. The API features an authentication method implemented with 'HTTPAUTH' for security.
3. In the backend, the machine learning code processes the image and returns a dictionary with key value pairs holding attributes: object name, accuracy, its classification type. The backend also updates the API with the processed information using a PUT request.
4. The backend file connects to the Mongo Atlas database and stores the classification and accuracy information processed from the Machine Learning code.
5. The RasPi receives information from the API, whether the object is 'trash' or 'recyclable'.
6. The RasPi outputs this signal to the stepper motor, which sorts the object.
7. The API then sends the information to the iOS application which presents interactive results to the user.

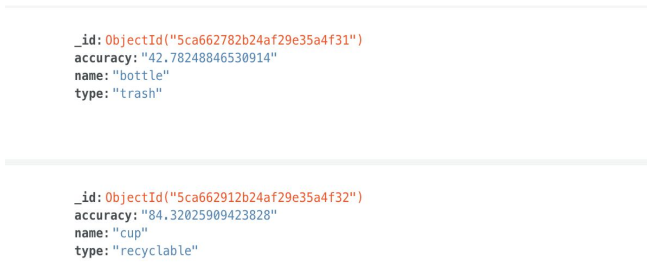
The user and recycling information is stored in a Mongo Atlas database. The technicalities behind the working of the database is as follows:



```
classify = [
    {
        'id': 1,
        'name': 'nameholder',
        'string': 'stringplaceholder',
        'type': 'Trash',
        'accuracy': 80,
        'recyclable': 0,
        'trash': 0,
        'accuracyavg': 0,
    }

]
```

→



```
_id: ObjectId("5ca662782b24af29e35a4f31")
accuracy: "42.78248846530914"
name: "bottle"
type: "trash"

_id: ObjectId("5ca662912b24af29e35a4f32")
accuracy: "84.32025909423828"
name: "cup"
type: "recyclable"
```

The image on the left shows how the information is stored in the API backend . The respective placeholders **'name'** , **'string'** , **'type'** , **'accuracy'**, **'recyclable'**, **'trash'**, **'accuracyavg'** are able to be securely accessed by GET and PUT requests from the main ecobin.py and backend.py files connected to the Pi, the Mongo database and iOS application.

The image on the right shows the final processed information stored in the database before the iOS application can access this data and display it for the user. Each item's classification is stored as a "document" in a "collection" inside a "database" within Mongo Atlas.

5. Cost Breakdown

The cost breakdown approximate costs to produce one Ecobin unit. It accounts for hardware, software, and mechanical components of the Ecobin. For software, all of the components including access to a cloud instance, have been negligible. For the mechanical components, the major items involve the motors, the motor controllers, and the materials to construct the Ecobin housing and platform.

Mr. Cootner, the primary customer, did not specify any components. There were also no items donated by the customer or any professors. The team was given flexibility with components as long as they fulfill the engineering requirements as specified in the Appendix.

Item	Description	Cost/Unit	Quantity	Total Cost
	Hardware Components			
1	Raspberry Pi 3 B+ (1.4 Ghz Cortex-A53 with 1 GB RAM)	$54.99	1	$54.99
2	DC 12V LED Strips	$8.95	1	$8.95
3	Raspberry Pi Camera Module V2-8 Megapixel, 1080p	$24.90	1	$24.90
4	PIR Motion Sensor	$9.95	1	$9.95
5	Breadboard PCB	$12.95	1	$12.95
6	2N2222 npn-BJT	$0.45	1	$0.45
7	Jumper Wires Pack	$2.00	1	$2.00
8	Wide Angle Lens	$12.00	1	$12.00
9	HC-SR04 Ultrasonic Sensor	$4.00	2	$8.00
10	Power Supply	$20.00	1	$20.00
	Mechanical Components			
11	Phidget Stepper Motor HC (Driver)	$90.00	1	$90.00
12	28STH32 NEMA - 11 Bipolar Stepper with 27:1 Gearbox	$36.00	1	$48.00
13	Power Supply 12V 2A	$10.00	1	$10.00
14	Mini-USB Cable	$4.00	1	$4.00
13	Stepper Motor (NEMA-17)	$14.00	1	$13.00
14	TB6612 Stepper Motor Driver	$4.95	1	$4.95
15	Arduino Uno R3	$20.00	1	$20.00
16	Trash Receptacle	$50.22	1	$50.22
17	Black HDPE	$64.75	1	$64.75
18	3D Printing Filament	$19.99	1	$19.99
19	Shaft, Pulley Teeth, Rod, Timing Belt	$25.00	1	$25.00
	Total Cost	-	-	$490.01

Table 5.1. Production Cost Estimate for Ecobin

The budget estimate comprises of two sections: hardware and mechanical components. Hardware components involve items that are directly associated with the major hardware, such as the RasPi and the sensors. Similarly, mechanical components involve items that are associated to the motors and the controllers.

For an R&D project, Ecobin is relatively inexpensive, standing at $490.01. In production, Ecobin will cost significantly less. The high cost was due to its usage of proprietary parts. If the Ecobin goes into mass production, process improvement will yield better operational efficiency in manufacturing: components would be purchased in bulk and each component can be optimized further. In addition, designing a specialized microcontroller instead of a Raspberry Pi or Arduino will also help improving cost leadership. It is noteworthy that no other smart trash can has this the capability to effectively and seamlessly classify trash versus recyclable, so the prototype price tag is reasonable.

6. Appendices

Appendix A - Specifications

Team 9
Team Name: Ecobin
Project Name: Ecobin

Requirements	**Value, range, tolerance, units**
Case Dimensions	1.5m x 0.6m x 0.6m
Power Supply	Single 110VAC wall outlet connection with outputs for ● 5V, 12W for Raspi ● 12V, 0.9W for LED strip ● As required for sorting mechanism and lid
Energy Use	<0.7 kWh/day (about the same as Energy Star rated compact refrigerators)
Object Detection	● Phase 1 - Successfully identify and sort two separate types of waste, recyclable plastic and general trash, at a 90% success rate. ● Phase 2 - Perform sorting tasks on three types of waste: Combustible, plastic, and general trash, all at a 90% success rate. ● Phase 3 - Completion of edge cases with a 90% success rate which include the following but are not limited to: ○ *Scenario A*: Plastic within a non-recyclable plastic waste (ie. Plastic bottle inside a paper bag) ○ *Scenario B*: User throws away waste that must be treated with caution (ie. lithium ion batteries, toxic waste [1]).
Electro-Mechanical Sorting Tool	● Sustain waste with the weight up to 1.0kg (2.2lbs). ● Move object to the desired compartment within 5s.
Practical	The overall cost of the final Ecobin design should be < $150.
Cloud Information	● Data processing for object classification via a cloud instance. Processing time < 2s. ● Desired specifications: 8 CPU cores, 16GB RAM, 10 GB storage.

	● Current specifications (per node on BU Shared Computing Cluster): 2 14-core CPUs, 256 GB RAM, 886 GB storage.
User Interaction	● Mobile app sync time of < 2s. ● Keeps track of user disposal history for the past 30 days.
Internet Connection	Minimum upload speed of 500 kbps required

Appendix B – Team Information

Team 9, or Team Ecobin, consists of Charles Thao, Shreenidhi Jayaram, Hayato Nakamura, Aditya Wikara, and Kevin Sinaga. The first three members are the team's computer engineers, while the last two members are electrical engineers. We chose to create Ecobin because we realized that despite the high rate of recyclables in circulation, recycling rates are low in American households. Recycling is an important solution to curb negative effects of climate change and global warming on the whole. By eliminating the inconvenience associated with the process, we hope to increase recycling rates worldwide and go green. In addition, the proposal for Ecobin posed very interesting engineering problems that involve many disciplines in software, hardware and mechanical engineering. Therefore, members of the team are intrigued and decided to pursue the project.

Aditya Wikara is a graduating Electrical Engineer who has a passion for many different subjects including clean technologies, energy, and IoT devices. He hopes to use his skills as an engineer to one day develop a startup and continue to solve important problems.

Shreenidhi Jayaram is a graduating Computer Engineer who has a passion programming, finance and technology. She hopes to use her skills as an engineer to work in the fintech industry and use technology in investment banking applications. Starting this Fall, Jayaram will work to ensure low latency trading at Deutsche Bank.

Charles Thao is a graduating Computer Engineer. Throughout his time at Boston University, he has been involved in projects in cloud computing, high performance computing, web development and and machine learning. He hopes to use his skills as an engineer to found a business and continue to work on interesting problems throughout his career. He will pursue a Master's degree at Boston University.

Hayato Nakamura is a graduating Computer Engineer who is passionate about app development and machine learning. He hopes to use his skills as an engineer to work on many more engineering projects focused on the same areas. He will pursue a Master's degree at Columbia University.

Kevin Sinaga is a graduating Electrical Engineer who has a passion for many different subjects including clean energy, IoT devices, smart electric power and much more. He hopes to use his skills as an engineer to work on solving more engineering problems related to sustainability and clean electric power.

Working alongside us in developing Ecobin are also 4 mechanical engineers: Esther Huynh, Jiaqian Wu, Qianqian Guo, and Yuzi Li.