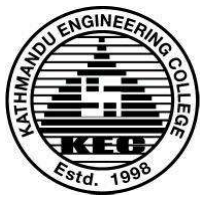
TRIBHUVAN UNIVERSITY
INSTITUTE OF ENGINEERING

Kathmandu Engineering College

Department of Computer Engineering



Major Final Report

On

"SAFA SAMAJ"-Waste Management System

[Code No: CT 755]

By

Ojaswi Kafle - KAT073BCT049

Rojan Adhikari - KAT073BCT061

Suhana Pradhan - KAT073BCT082

Kathmandu, Nepal

FALGUN, 2077

TRIBHUVAN UNIVERSITY

INSTITUTE OF ENGINEERING

Kathmandu Engineering College

Department of Computer Engineering

"SAFA SAMAJ"-Waste Management System

[Code No: CT 755]

PROJECT REPORT SUBMITTED TO

THE DEPARTMENT OF COMPUTER ENGINEERING

IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR

THE BACHELOR OF ENGINEERING

By

Ojaswi Kafle - KAT073BCT049

Rojan Adhikari - KAT073BCT061

Suhana Pradhan - KAT073BCT082

Kathmandu, Nepal

FALGUN, 2077

TRIBHUVAN UNIVERSITY

Kathmandu Engineering College

Department of Computer Engineering

TRIBHUVAN UNIVERSITY
INSTITUTE OF ENGINEERING

Kathmandu Engineering College
Department of Computer Engineering

CERTIFICATE

The undersigned certify that they have read and recommended to the Department of Computer Engineering, a major project work entitled "Safa Samaj- Waste Management System" submitted by Ojaswi Kafle – KAT073BCT049, Rojan Adhikari– KAT073BCT061, Suhana Pradhan – KAT073BCT082 in partial fulfillment of the requirements for the degree of Bachelor of Engineering.

Sudeep Shakya
(Head of Department)
(Project Supervisor)
Department of Computer Engineering
Kathmandu Engineering College

Prof. Dr Subarna Shakya
(External Examiner)
Professor
Institute of Engineering (IOE)
Pulchowk Campus

Er. Sapana Thakulla
(Project Coordinator)
Department of Computer Engineering
Kathmandu Engineering College

COPYRIGHT

ACKNOWLEDGEMENT

We would like to extend our cordial gratitude to **Institute of Engineering (IOE)** for providing us with the opportunity to develop a project under the academic requirement as a major project. We are very grateful to the **Department of Computer Engineering, Kathmandu Engineering College** for helping us evolve our programming techniques. This project will help us develop our skills related to programming and management skills for future references. We are also very grateful to our project supervisor and the **Head of Department of Computer Engineering, Er. Sudeep Shakya** and **Deputy Head of Computer Engineering, Er. Kunjan Amatya** for their scholarly guidelines and suggestions during the development of our project. We also acknowledge the help and support provided by our project coordinator, **Er. Sapana Thakulla**. Without their guidance, it would be an uphill task.

So finally we would give our sincere thanks to all those who were involved in any which way in our project and are grateful for your contribution.

ABSTRACT

The purpose of this project is to develop an app "**Safa Samaj**", more particularly, a waste management app whose motive is the proper and systematic waste management resulting more convenience to the general public. This system requires user's as well as driver's details, ultimately then saved in our admin's database, and the user will have familiarization with the event concluding the date of degradable and non-degradable wastes collection as well as indication of their location and then notify the drivers. Drivers on the other hand, gets notified from users, view the events as well as location/destination on the map and follows the route to pick up the waste from the destination and are further processed or completely disposed. Drivers are required to enter their license details as well as users details are also needed to be entered and are kept highly confidential. The users are provided with systematic management service of their daily generated wastes. For waste management after collection, image classification is done using keras framework and CNN.

Keywords: android app, software application, waste management, location, image classification

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

ADT : Android Development Tools

APK : Android Application Package

CNN : Convolutional Neural Network

FCM : Firebase Cloud Messaging

HTTP : Hyper Text Transfer protocol

IDE : Integrated Development Environment

JVM : Java virtual machine

RCRA : Resource Conservation and Recovery Act

SCA : Service Corporation of America

SDLC : Software Development Life Cycle.

WORA : Write once run anywhere

CHAPTER 1: INTRODUCTION

1.1 BACKGROUND THEORY

Waste management is one of the burning issue that has emerged as a global crisis everywhere in the world. Around the world, waste generation rates are rising. In 2016, the worlds' cities generated 2.01 billion tons of solid waste, amounting to a footprint of 0.74 kilograms per person per day. With rapid population growth and urbanization, annual waste generation is expected to increase by 70% from 2016 levels to 3.40 billion tons in 2050. Managing solid waste is one of the major challenges in urbanization. In context to our country Nepal, a survey conducted in all 58 municipalities in 2012 found that the average municipal solid waste generation was 317 grams per capita per day. This translates into 1,435 tons per day or 524,000 tons per year of municipal solid waste generation in Nepal. Many of these technically and financially constrained municipalities are still practicing roadside waste pickup from open piles and open dumping, creating major health risks [1].

Managing waste properly is essential for building sustainable and livable cities, but it remains a challenge for many developing countries and cities. Effective waste management is expensive, often comprising 20% – 50% of municipal budgets. Operating this essential municipal service requires integrated systems that are efficient, sustainable, and socially supported.

1.2 PROBLEM STATEMENT

In the present context, the waste generated by each house is collected by the municipal workers twice a week. These workers collect all the waste on the single container/vehicle and hence, the separation of the bio-degradable and non-biodegradable wastes are not distinguished. Finally the dumpster is dumped into the river sides or away from the residence area. This unusual schedule for the waste management is causing a serious issues, such as:

- No proper mechanism differentiate between biodegradable and non-biodegradable wastes.

- There is no proper time-schedule for the municipal workers to come at work.
- Terrific stench of garbage over some period of time causing infections.

1.3 OBJECTIVES

The major objectives are as follows:

- To properly manage of biodegradable and non-biodegradable waste, as they are later classified during management process.
- To provide notification to the drivers to collect the wastes in the located regions.
- To provide better facilities of maps.

1.4 SCOPE AND APPLICATION

This application can be used by everyone that is aware about protecting the environment. Some of the major applications are as follows:

- Making the society clean.
- Proper disposal of waste.
- Map for driver to reach the destination.
- Public and private partnerships offers opportunities for operational efficiency and cost effectiveness.

CHAPTER 2: LITERATURE REVIEW

Solid waste has been long recognized as a potential problem worldwide that deserves serious attention, Resource Conservation and Recovery Act (RCRA) [2]. Solid Waste states that "solid waste" means any garbage or refuse, sludge from a wastewater treatment plant, water supply treatment plant, or air pollution control facility and other discarded material, resulting from industrial, commercial, mining, and agricultural operations, and from community activities. Nearly everything we do leaves behind some kind of waste. It is important to note that the definition of solid waste is not limited to wastes that are physically solid. Many solid wastes are liquid, semi-solid, or contained gaseous material.

According to prokerala the types of solid waste may be divided into different types of waste and that depends upon their source [3]. Broadly the types of solid waste include: Household waste mostly consists of household waste, sanitation waste, waste from streets, demolition debris that arises during the construction and demolition of buildings and other construction activities. With the increase in the urbanization, municipal solid waste is forming the bulk part of solid waste. The growth of metropolitan cities is even leading to an enormous amount of municipal solid waste. Industrial waste is a waste that is quite dangerous as they consist of toxic substances that are of chemical nature. This type of waste is highly dangerous to human, plants, animals and the overall environment. As improper disposal of the industrial solid waste may lead to death, disease and sometimes.an environmental damage that may continue for generations. For example: any oil spill in the seas, oceans or release of poison gases, chemicals in the air and improper disposal of industrial effluents into the soil will lead to destruction of all living species in addition to environmental damage. Hospital waste or Biomedical wastes are other form of solid waste that is being generated day in day out by various hospitals, clinics, research centers, pharmaceutical companies and health care centers. This type of solid waste is most infectious and can spread diseases and other types of viral and bacterial infections among humans and animals if not managed properly in a scientific way. The hospital waste includes solid waste in the form of disposable syringes, bandages, cotton swabs, body fluids, human excreta, anatomical waste, bandages, expired medicines, and other types of

chemical and biological waste. Hospital waste is equally hazardous and dangerous as in case of industrial waste if not disposed off or managed properly.

Managing solid waste is one of the major challenges in urbanization. High population growth rate and increase of economic activities in the urban areas of developing countries combined with the lack of training in modern solid waste management practices complicate the efforts to improve the solid waste management services which has been a huge problem in people's day to day life and that has been affecting human beings directly or indirectly.

A survey conducted in all 58 municipalities of Nepal in 2012 found that the average municipal solid waste generation was 317 grams per capita per day. This translates into 1,435 tons per day or 524,000 tons per year of municipal solid waste generation in Nepal. Many of these technically and financially constrained municipalities are still practicing roadside waste pickup from open piles and open dumping, creating major health risks. Kathmandu valley has only around 1300 staffs as solid waste management personals that is about 60% of overall municipal staff. Instead of that there are lots of problem regarding solid waste management in Kathmandu valley.
Currently 90% of the total generated waste in Kathmandu is collected by Roadside collection, door-to-door collection and container collection method.

Many U.S. communities now actively recycle. Common programs include recycling containers. The community provides containers in which individual families deposit such materials as newspapers; glass bottles and jars; tin and aluminum containers; plastic bottles and bags; mixed waste paper (cardboard, phone books, magazines, junk mail, office paper, brown bags) and used motor oil. The community arranges for curbside pickup and delivery to a recycling facility. Drop-off recycling zones. Groups of large recycling bins are installed on public property in one or more locations throughout the community. Curbside Recycling centers. The community provides the center itself and encourages residents to drop off or sell refuse materials there. Green waste diversion and composting programs. Leaves, grass clippings, and other organic waste materials are composted and used to enrich soil or as mulch or landfill cover. Regardless of the specific techniques employed,

it has been clear that the best outcomes are achieved with individual role and formal program on solid waste management.

In 1893, Harm Huizenga, a Dutch immigrant, began hauling garbage at $1.25/wagon in Chicago. In 1968, Wayne Huizenga, Dean Buntrock, and Larry Beck founded Waste Management, Inc. and began aggressively purchasing many of the smaller garbage collection services across the country. In 1971, Waste Management went public, and by 1972, the company had made 133 acquisitions with $82 million in revenue. It had 60,000 commercial and industrial accounts and 600,000 residential customers in 19 states and the provinces of Ontario and Quebec. In the 1980s, Waste Management acquired Service Corporation of America (SCA) to become the largest waste hauler in the country. In 1993, Waste Management, Inc. changes to WMX Technologies, Inc [4]. As a universal symbol of the other services they provided other than solid waste removal, recycling, and disposal. Municipal solid waste management problem [5].

In the context of Nepal, Khalisisi is a websites whose vision is to standardize proper waste management and implement it across the country such that 90% of the waste we generate would be recycled instead of ending up in landfills. Their main mission is to gap the bridge between waste entrepreneurs and waste sellers [6].

The other very popular website on waste management in Nepal is Doko Recyclers [7]. They segregate the dry waste they've collected and separately package paper, plastics and metals to sell them directly to their respective factories. These initiatives have been a huge step taken for the betterment of our country.

CURRENT SITUATION:

In the context of our country, the waste are collected in the interval of 3 days or even in the interval of a week. As the interval of a week is a longtime span which results in the collection of large amount of waste from each homes. There are also situations when the people miss to throw the wastes when they are bring collected due to their daily schedule. Other problems may arise if the waste is collected today but the particular house have a party tonight resulting in the huge amount of waste to be collected which adds up the waste plus the waste from each day. So our application aims to minimize this problem as the user can notify the drivers whenever the waste needs to be picked up when the dustbin is full. Also in our country, there is no proper disposal for the degradable and non-degradable wastes resulting in the pollution of the environment.

To be exact around 800 tons to 1000 tons of wastes are collected in Nepal per day. Among these wastes around 90% can be recycled. Currently, some municipal government is providing a trash collection service and people have to pay Rs.200 per month to display these trash. It is also calculated that around 11.5% of people are below the range who are unable to pay this amount to government just to dispose wastes. In present context, there is no proper waste management system in our country but only just “Sweep and Dump” as well as burn the trash to clean up.

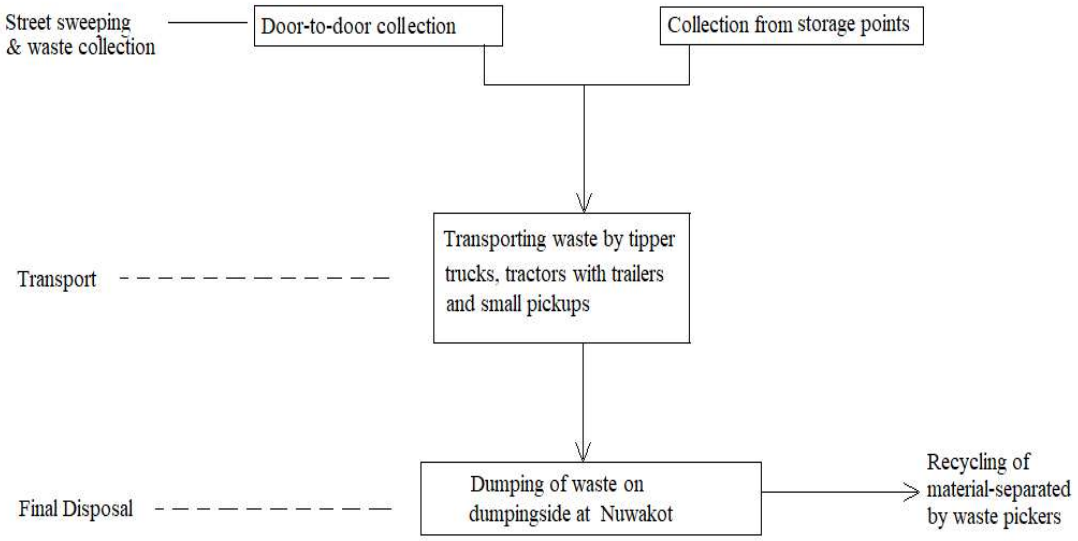


Figure 2.1: Present scenario of waste management in Nepal

Total waste(gm)	Organic	Paper	Plastic	Metal	Inert	Glass	Textile
63520.00	47,818	2,714	5,998	92	4,874	1,687	349
100%	75.28%	4.27%	9.43%	0.14%	7.67%	2.66%	0.55%

Figure 2.2: Residential Waste Production per month

CHAPTER 3: METHODOLOGY

3.1 METHODOLOGY

3.1.1 DATA COLLECTION

Our project consists of two parts: App and Image Processing.

For the app part, data collection is done from users, drivers, and admin. From user data collection is done as User's name, Address, Email, no. of family members, house number and phone number. From driver data collection is done as Driver's name, Address, Email, License Number and Phone number. For admin, data collection is done as Name, Address, Email, No. of a family member (if using), House number (if using) and License no. (if driver). Admin has authority to add user, driver, and other admin based upon these data collection.

For the image processing part, our data set is derived from Kaggle. The majority of our datasets are Kaggle datasets. This includes the training data set. In the case of the validation data set, we have included the images clicked by ourselves too. (Total dataset includes two folders: Training and Validation)

3.1.2 CONVOLUTION NEURAL NETWORK (CNN)

In neural networks, Convolutional neural network (ConvNets or CNNs) is one of the main categories to do images recognition, images classifications. Objects detections, recognition faces etc., are some of the areas where CNNs are widely used. CNN image classifications takes an input image, process it and classify it under certain categories. Technically, deep learning CNN models to train and test, each input image will pass it through a series of convolution layers with filters (Kernals) [9].

Along with filters (Kernals), Pooling, fully connected layers (FC) and Softmax function are applied to classify an object with probabilistic values between 0 and 1. The below figure is a complete flow of CNN to process an input image and classifies the objects based on values [10].

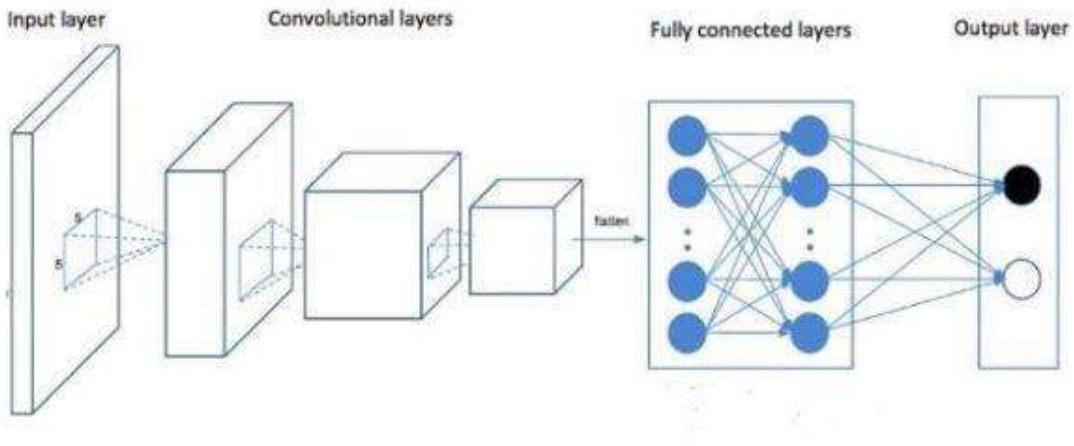


Fig 3.1: Convolutional Neural Network

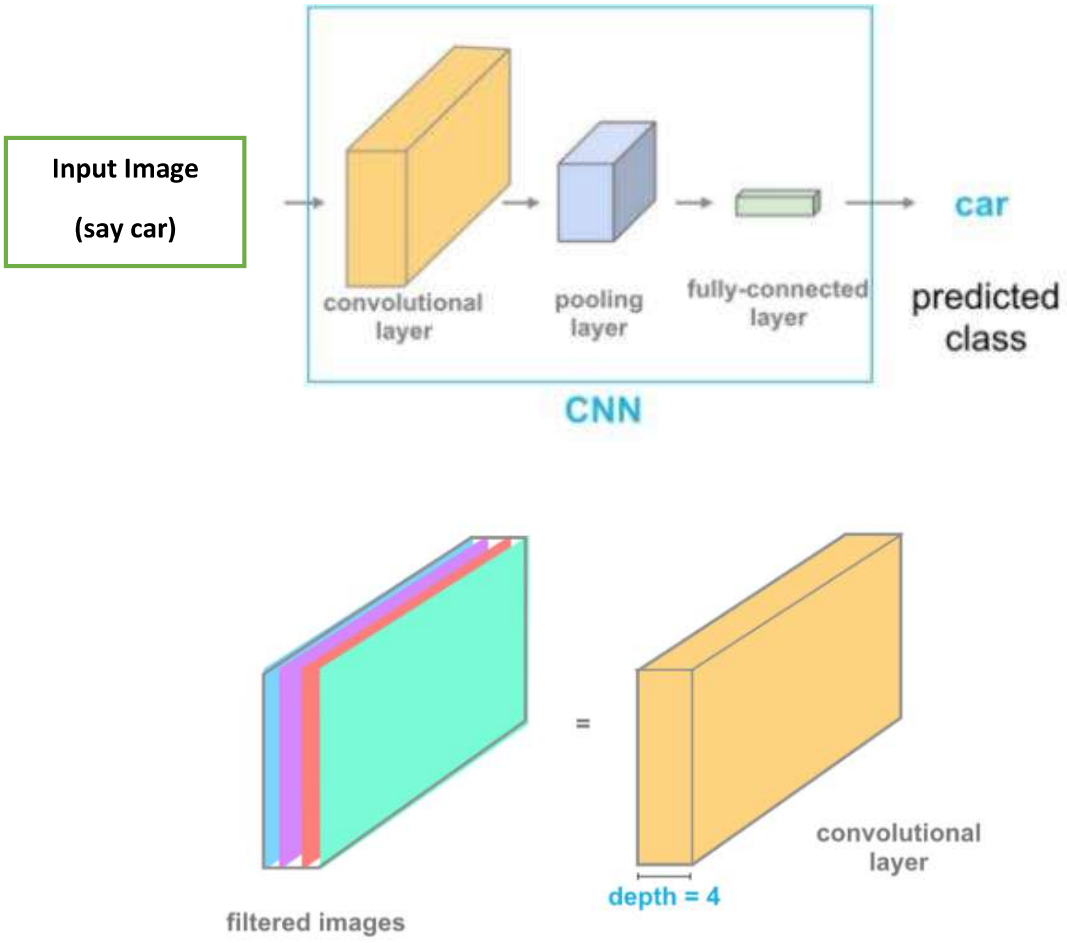
Convolution is the first layer to extract features from an input image. Convolution preserves the relationship between pixels by learning image features using small squares of input data. It is a mathematical operation that takes two inputs such as image matrix and a filter or kernel.

Working of CNN Model:

In CNN, any image represent some pixels. Nearby pixels in an image can be analysed by using a filter *(*can be called as weights, kernels or features**)**

Filters are tensor which keeps track of spatial information and learns to extract features like edge detection, smooth curve, etc of objects in a convolutional layer. The major part is to

detect edges in the images and these are detected by the filters. It helps to filter out unwanted information to amplify images.



Let's consider image of size 5X5 size and 3 filters (since each filter will be used for each color channel: RGB) of 3X3 size. For simplicity, we took 0 and 1 for filters, usually, these are continuous values. The filter convolute with the image to detect patterns and features.

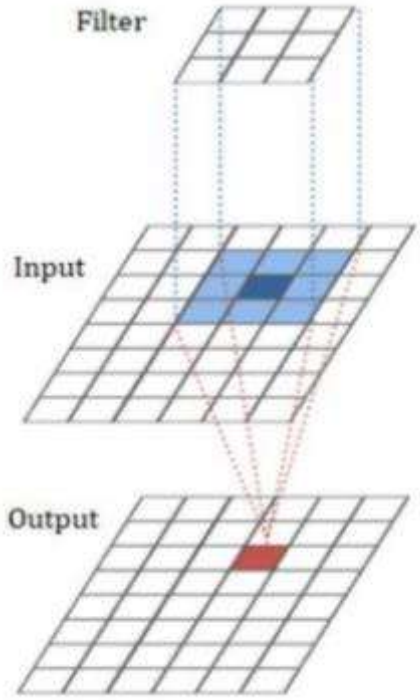
Ouput image value = LRF * Filter
(dot product of LRF and Filter)

Filter size = 3 X 3 --> 3
Input size = 5 X 5 --> 5
Stride = 1X1-->1 (1 cell move)
Padding = 0X0-->0 (No padding)

output size = (Input size - Filter size + 2 * Padding)* Stride + 1

output size = $\boxed{(I - F + 2P) * S + 1}$

output size = (5 - 3) * 1 + 1
output size = 3 -->3 X 3



Then the convolution of 5 x 5 image matrix multiplies with 3 x 3 filter matrix which is called **"**Feature Map**"**. Dot product is applied to the scaler value and then filter is moved by the stride over the entire image.

Sometimes filter does not fit perfectly fit the input image. Then there is a need to pad the image with zeros as shown below. This is called padding**.**

Next, the size of images is reduced , if they are too large. Pooling layers section would reduce the number of parameters when the images are too large. Padding is applied so that the filter perfectly fits the given image. Adding pooling layer then decrease the size of the image and hence decrease the complexity and computations.

Next Step, is Normalization**.** Usually, an activation function ReLu is used. ReLU stands for Rectified Linear Unit for a non-linear operation.

The output is $f(x) = max(0,x)$.

The purpose of ReLu is to add non-linearity to the convolutional network. In usual cases, the real-world data want our network to learn non-linear values.

A rectified linear unit has output 0 if the input is less than 0, and raw output otherwise. That is, if the input is greater than 0, the output is equal to the input. Here we are assuming that we have negative values since dealing with the real-world data. In case, if there is no negative value, you can skip this part.

RELU(X)= {0 if x<0

x if x >=0 }

As the final step flatten the matrix and feed the values to fully connected layer.

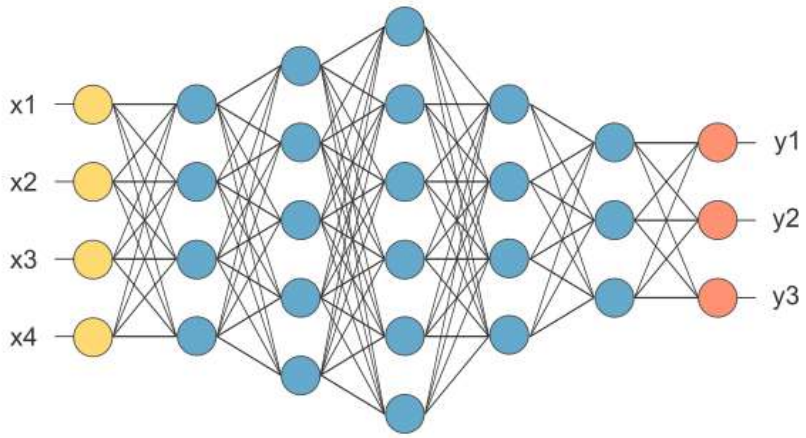
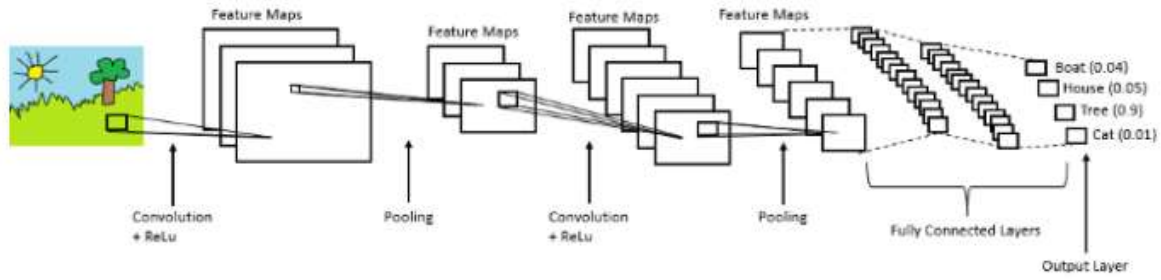


Fig: Convolution Neural Network

Next, we need to train the model in the same way, we train other neural networks. Using the ceratin number of epochs and then backpropagate to update weights and calculate the loss.

Overall Structure of CNN



Hence, Convolution Neural Networks involves following processes:

1. Convolution layer where convolution happens.
2. Pooling layer where the pooling process happens
3. Normalization usually with the use of ReLu
4. Fully Connected Layers

3.1.3 ALGORITHM

3.1.3.1 APPLICATION ALGORITHM

Step 1: Start.

Step 2: If user then

Register and login with the user id and goto step 3.

Else

Login with driver id and goto step 6.

Step 3: Disposal of waste.

Step 4: Pin the location from where the waste must be collected.

Step 5: Send notification to the driver.

Step 6: View the location that is send by the user.

Step 7: Use the map to reach the destination.

Step 8: Collect the waste from the destination.

Step 9: Unpin the location from where the waste is collected.

Step 10: Management of the collected waste.

Step 11: Stop.

3.1.3.2 ALGORITHM FOR CONVOLUTION NEURAL NETWORK

- Provide input image into convolution layer
- Choose parameters, apply filters with strides, padding if requires. Perform convolution on the image and apply ReLU activation to the matrix.
- Perform pooling to reduce dimensionality size
- Add as many convolutional layers until satisfied
- Flatten the output and feed into a fully connected layer (FC Layer)
- Output the class using an activation function (Logistic Regression with cost functions) and classifies images.

3.1.4 TOOLS USED

3.1.4.1 Android Studio:

Android Studio is the official integrated development environment (IDE) for Android application development. It is based on the IntelliJ IDEA, a Java integrated development environment for software, and incorporates its code editing and developer tools.

To support application development within the Android operating system, Android Studio uses a Gradle-based build system, emulator, code templates, and Github integration. Every project in Android Studio has one or more modalities with source code and resource files. These modalities include Android app modules, Library modules, and Google App Engine modules.

Android Studio uses an Instant Push feature to push code and resource changes to a running application. A code editor assists the developer with writing code and offering code completion, refraction, and analysis. Applications built in Android Studio are then compiled into the APK format for submission to the Google Play Store.

The software was first announced at Google I/O in May 2013, and the first stable build was released in December 2014. Android Studio is available for Mac, Windows, and Linux desktop platforms. It replaced Eclipse Android Development Tools (ADT) as the primary IDE for Android application development [11].

3.1.4.2 Java:

Java is a popular general-purpose programming language and computing platform. It is fast, reliable, and secure. According to Oracle, the company that owns Java, Java runs on 3 billion devices worldwide. Java is a general-purpose programming language that is class-based, object-oriented, and designed to have as few implementation dependencies as possible. It is intended to let application developers write once, run anywhere (WORA), meaning that compiled Java code can run on all platforms that support Java without the need for recompilation. Java applications are typically compiled to bytecode that can run on any Java virtual machine (JVM) regardless of the

underlying computer architecture. As of 2019, Java was one of the most popular programming languages in use according to GitHub particularly for client-server web applications, with a reported 9 million developers [12].

3.1.4.3 Firebase:

Firebase is a mobile and web application development platform developed by Firebase, Inc. in 2011, then acquired by Google in 2014. It has services like Firebase Cloud Messaging (FCM) which is a cross-platform solution for messages and notifications for Android, iOS, and web applications, which as of 2016 can be used at no cost. Firebase Auth is a service that can authenticate users using only client-side code. It supports social login profiles Facebook, GitHub, Twitter and Google (and Google Play Games) [13].

Firebase provides a real time database and backend as a service. The service provides application developers an API that allows application data to be synchronized across clients and stored on Firebase's cloud. The company provides client libraries that enable integration with Android, iOS, JavaScript, Java, Objective-C, Swift and Node.js applications. The database is also accessible through a REST API and bindings for several JavaScript Frameworks such as Angular.js, React, Ember.js and Backbone.js. The REST API uses the Server-Sent events protocol, which is an API for creating HTTP connections for notifications from a server. Developers using the real time database can secure their data by using the company's server-side-enforced security rules.

3.1.4.4 Python:

Python is an interpreted, high-level and general-purpose programming language. Python's design philosophy emphasizes code readability with its notable use of significant whitespace. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects.

Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly, procedural), object-oriented and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library.

Python interpreters are supported for mainstream operating systems and available for a few more (and in the past supported many more). A global community of programmers develops and maintains CPython, a free and open-source reference implementation.

3.1.4.5 Keras:

Keras is a high-level neural networks library, written in Python and capable of running on top of either TensorFlow or Theano. It was developed with a focus on enabling fast experimentation. Being able to go from idea to result with the least possible delay is key to doing good research. It allows for easy and fast prototyping (through total modularity, minimalism, and extensibility).It supports both convolutional networks and recurrent networks, as well as combinations of the two. It supports arbitrary connectivity schemes (including multi-input and multi-output training). It runs seamlessly on CPU and GPU [14].

In keras framework we have imported ModelCheckpoint library to callback API and checkpoint the model weights. EarlyStopping is imported to find out the accuracy alter or decrease in particular epoch and stop the training at once to improve the performance. CSVLogger library is used to plot the training and validation loss and accuracy and represents as a string. Conv2D is used as the CNN model to filter the images by creating a convolution kernel. Flatten library is imported to reshape the tensor in a batch dimension. MaxPooling2d operation is used to calculate the maximum value in each batch of each feature map. It is applied by adding MapPooling2D layer provided by Keras API. Dense is the no. of layers expected as output. Dropout sets the input value to 0 and prevents overfitting. SpatialDropOut2D is used behind ConvLayer and drops entire 2D feature maps instead of individual elements. Sequential model is used as linear stack of different input, hidden and output layers. ImageDataGenerator is used to generate a data class for Image Augumentation. img_to_array converts PIL image instance to numpy array. array_to_img converts 3D numpy array to PIL image instance. load_img is used to load image in a PIL format.

3.1.4.6 Numpy:

NumPy is a Python library that provides a simple yet powerful data structure: the n-dimensional array. This is the foundation on which almost all the power of Python's data science toolkit is built, and learning NumPy is the first step on any Python data scientist's journey. NumPy is the fundamental package for scientific computing with Python. It contains among other things:

- a powerful N-dimensional array object
- sophisticated (broadcasting) functions
- tools for integrating C/C++ and Fortran code
- useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined. This allows NumPy to seamlessly and speedily integrate with a wide variety of databases

3.1.4.7 Matplotlib:

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shells, the Jupyter notebook, web application servers, and four graphical user interface toolkits. Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatterplots, etc., with just a few lines of code.

3.1.4.8 Tensorflow:

TensorFlow is a free and open-source software library for machine learning. It can be used across a range of tasks but has a particular focus on training and inference of deep neural networks. Tensorflow is a symbolic math library based on dataflow and differentiable programming. It is used for both research and production at Google.

TensorFlow was developed by the Google Brain team for internal Google use. It was released under the Apache License 2.0 in 2015. TensorFlow offers multiple levels of abstraction so you can choose the right one for your needs. Build and train models by using the high-level Keras API, which makes getting started with TensorFlow and machine learning easy. TensorFlow gives you the flexibility and control with features like the Keras Functional API and Model Subclassing API for creation of complex topologies. For easy prototyping and fast debugging, use eager execution.

TensorFlow also supports an ecosystem of powerful add-on libraries and models to experiment with, including Ragged Tensors, TensorFlow Probability, Tensor2Tensor and BERT [15].

3.1.4.9 OpenCV

OpenCV is a huge open-source library for computer vision, machine learning, and image processing. OpenCV supports a wide variety of programming languages like Python, C++, Java, etc. It can process images and videos to identify objects, faces, or even the handwriting of a human. When it is integrated with various libraries, such as Numpy which is a highly optimized library for numerical operations, then the number of weapons increases in your Arsenal i.e whatever operations one can do in Numpy can be combined with OpenCV.

This OpenCV tutorial will help you learn the Image-processing from Basics to Advance, like operations on Images, Videos using a huge set of OpenCV-programs and projects.

3.2 DEVELOPMENT METHODOLOGY

3.2.1 PROCESS MODEL

In order to implement our project, we decided to use iterative method for software development. Iterative process begins with a modest implementation of a subset of the software requirements and iteratively improves the developing versions until the full system is applied. At each iteration, design alterations are made and new functional competences are added. The simple idea behind this method is to develop a system through repeated cycles (iterative) and in smaller portions at a time. In this SDLC model, software development is divided into smaller segments to make the work easier. It focuses on initial, simplified implementation which becomes more complex after each iteration and features are added until the final product is deployed. Iterative method is also associated with an incremental model, so an early prototype is delivered to the client to see whether he/she want some changes or not. Thus, after each iteration a better version of product is delivered [8]. We are a group of three members, so we have chosen this model as it is easier to divide the work among us. A working software is generated quickly such that less time is spent on the documentation and more time is given for designing. We can detect errors at the early stages and correct it. Therefore, this model will be better for our project.

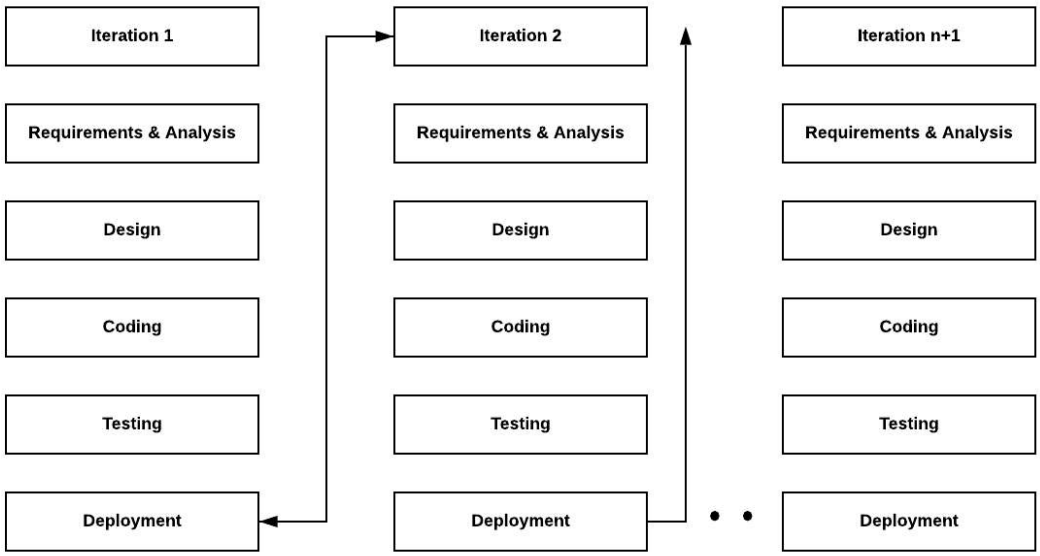


Figure 3.2: Iterative Model

3.2.2 SYSTEM BLOCK DIAGRAM

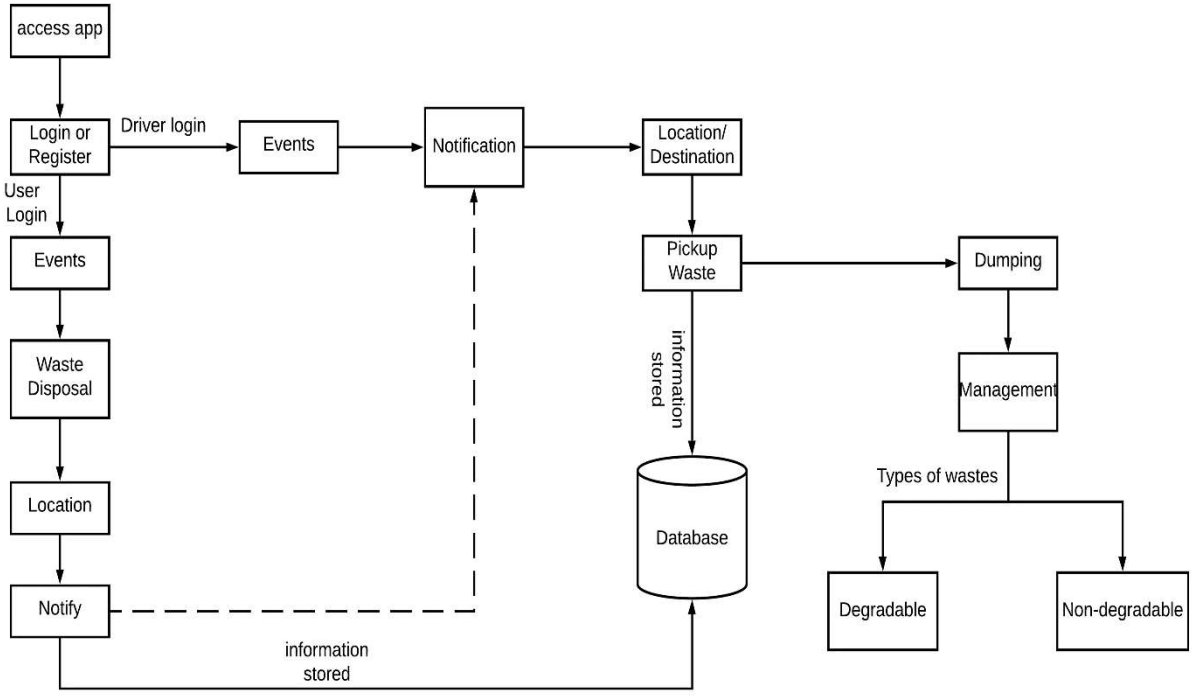


Figure 3.3: Block Diagram

Initially, the app is accessed. The person accessing the app may either be user or driver. The user will register/login to the system, view the calendar events and find out the day of collection of wastes. They can throw the wastes depending on the mentioned events in the calendar along with locating their location in the map. Then the driver is notified by the users to pick up the waste when dustbin is full.

Driver on the other side also login to the system. Their details are initially registered by admin themselves regarding validation and security issue. Driver can also view the events pinned on the calendar gets notifications and views the location/ destination to be reached.

After acquiring details about events, locations of the dustbin, they follow the map location covering user demand and pick up the wastes. They take the wastes for further processing where wastes are classified as degradable and non-degradable (plastic, cardboard, paper, glass, metal). Classification of types of wastes facilitates the proper management removing the current situation of clean and throw mechanism.

These details of users, location of the users, notifications, driver's details, information about waste pickup(location) are all being stored on the database which is ultimately in control of admin.

3.2.3 FLOWCHART

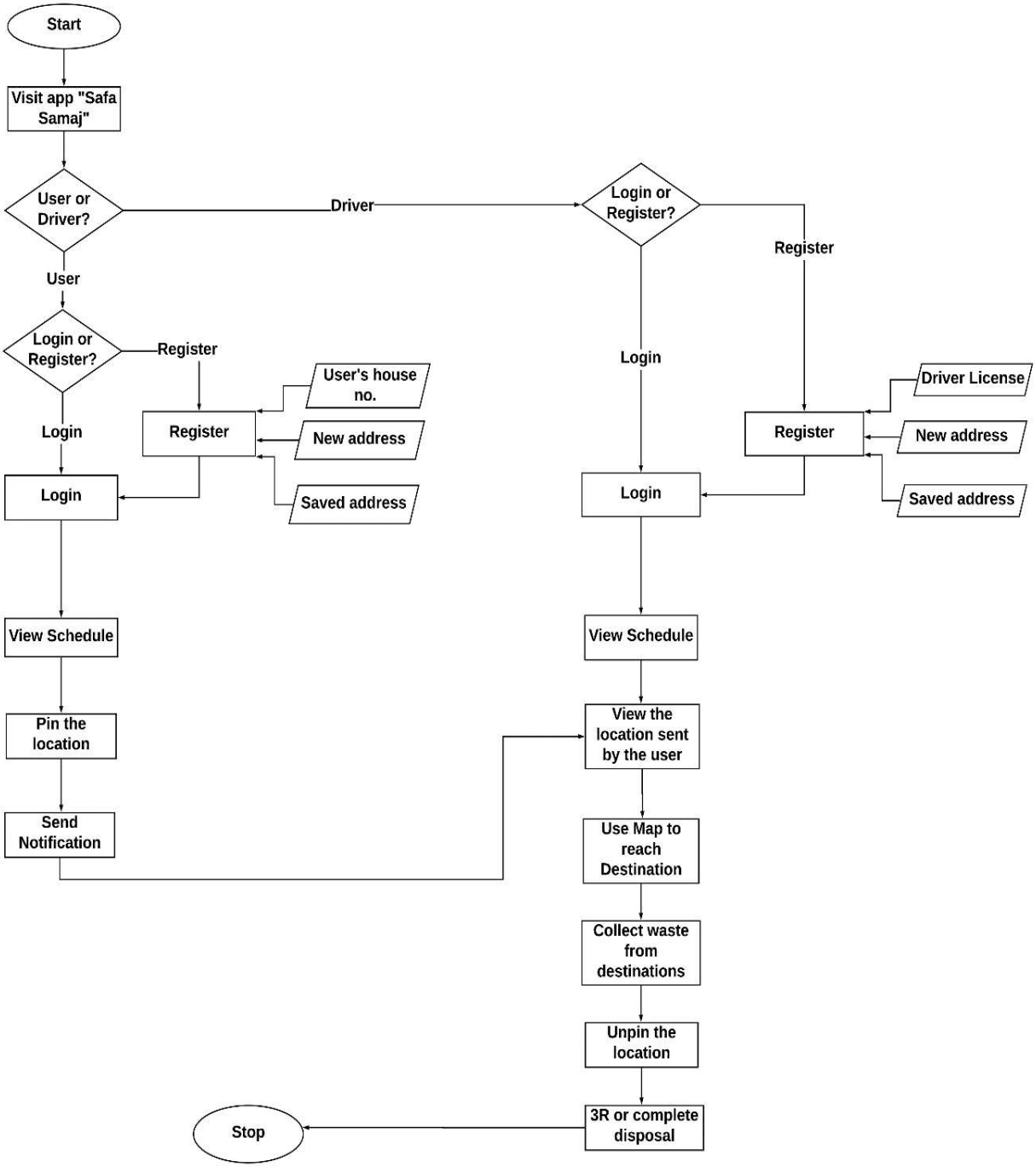


Figure 3.4: Flowchart

3.2.4 ENTITY RELATION DIAGRAM

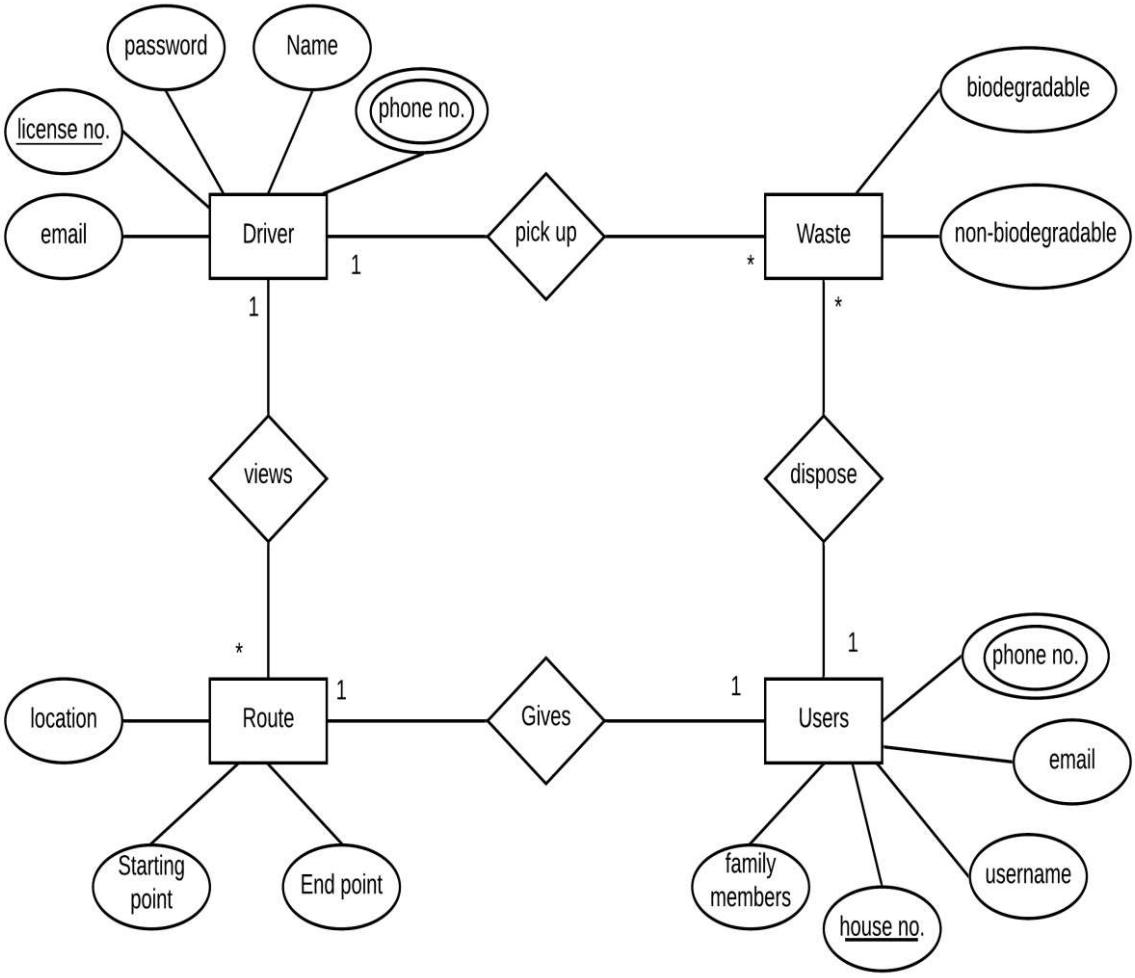


Figure 3.5: Entity Relationship Diagram

3.2.5 DATA FLOW DIAGRAMS

3.2.5.1 DFD level 0:

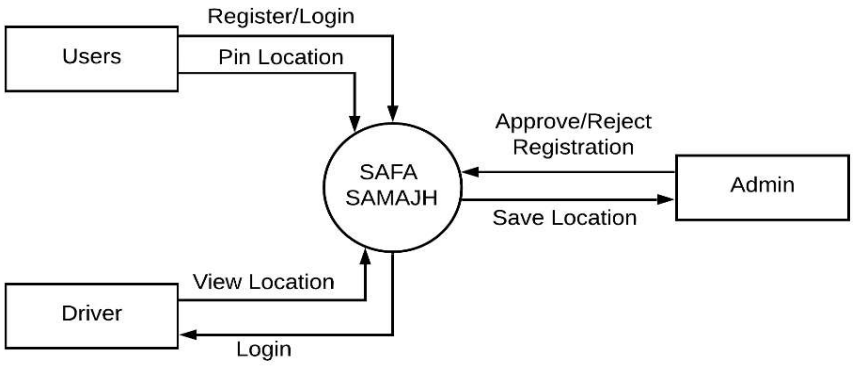


Figure 3.6: DFD level 0

3.2.5.2 DFD level 1:

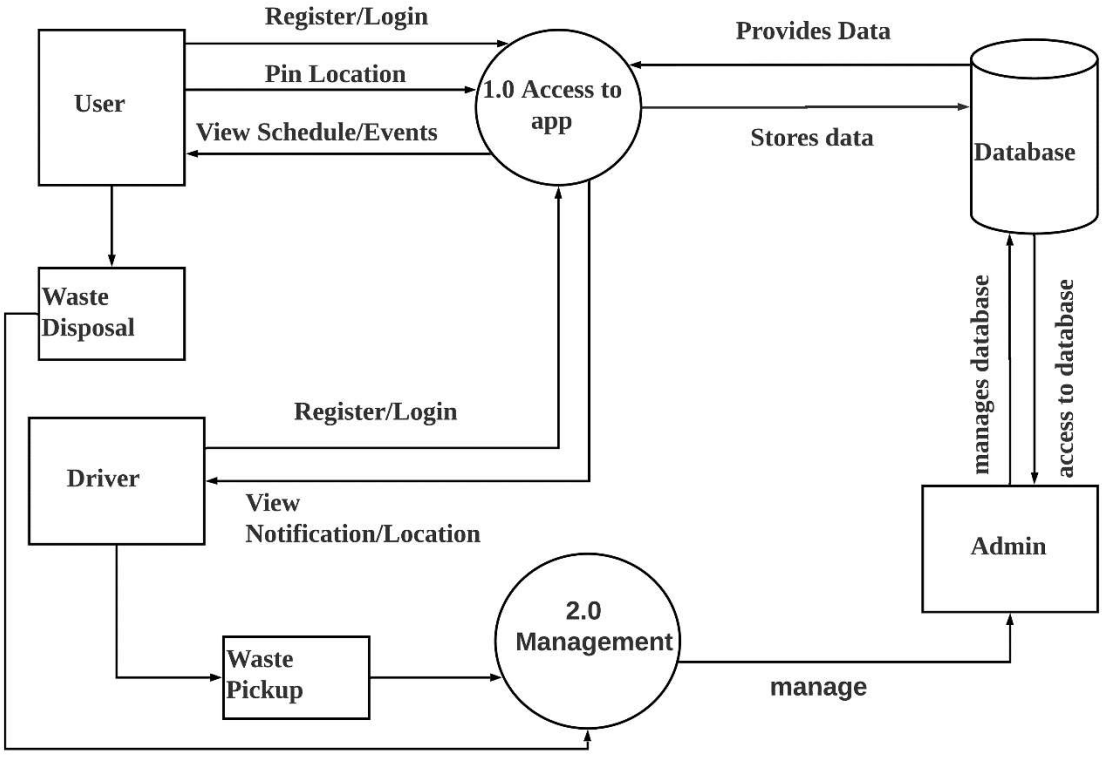


Figure 3.7: DFD level 1

3.2.5.3 DFD level 2:

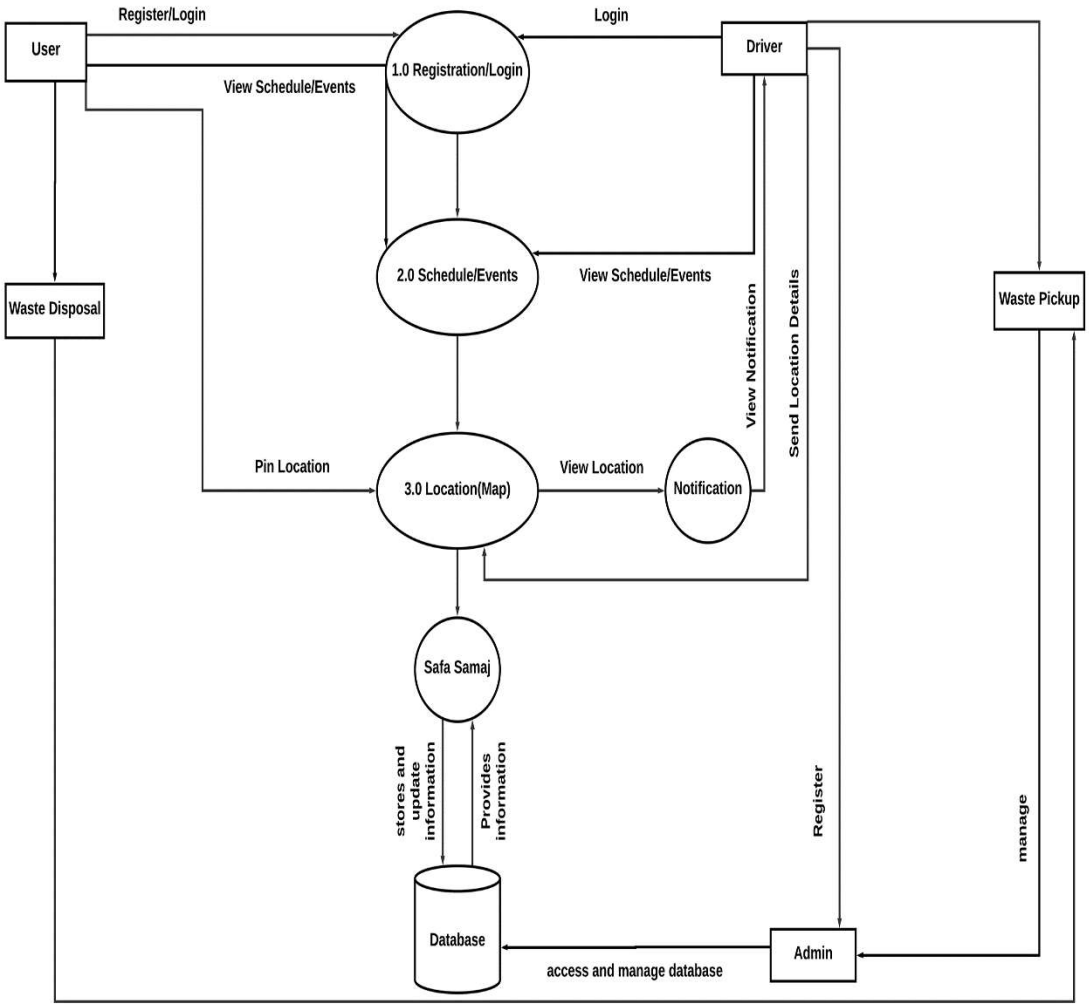


Figure 3.8: DFD level 2

3.2.6 UML Diagrams

3.2.6.1 USE CASE DIAGRAM

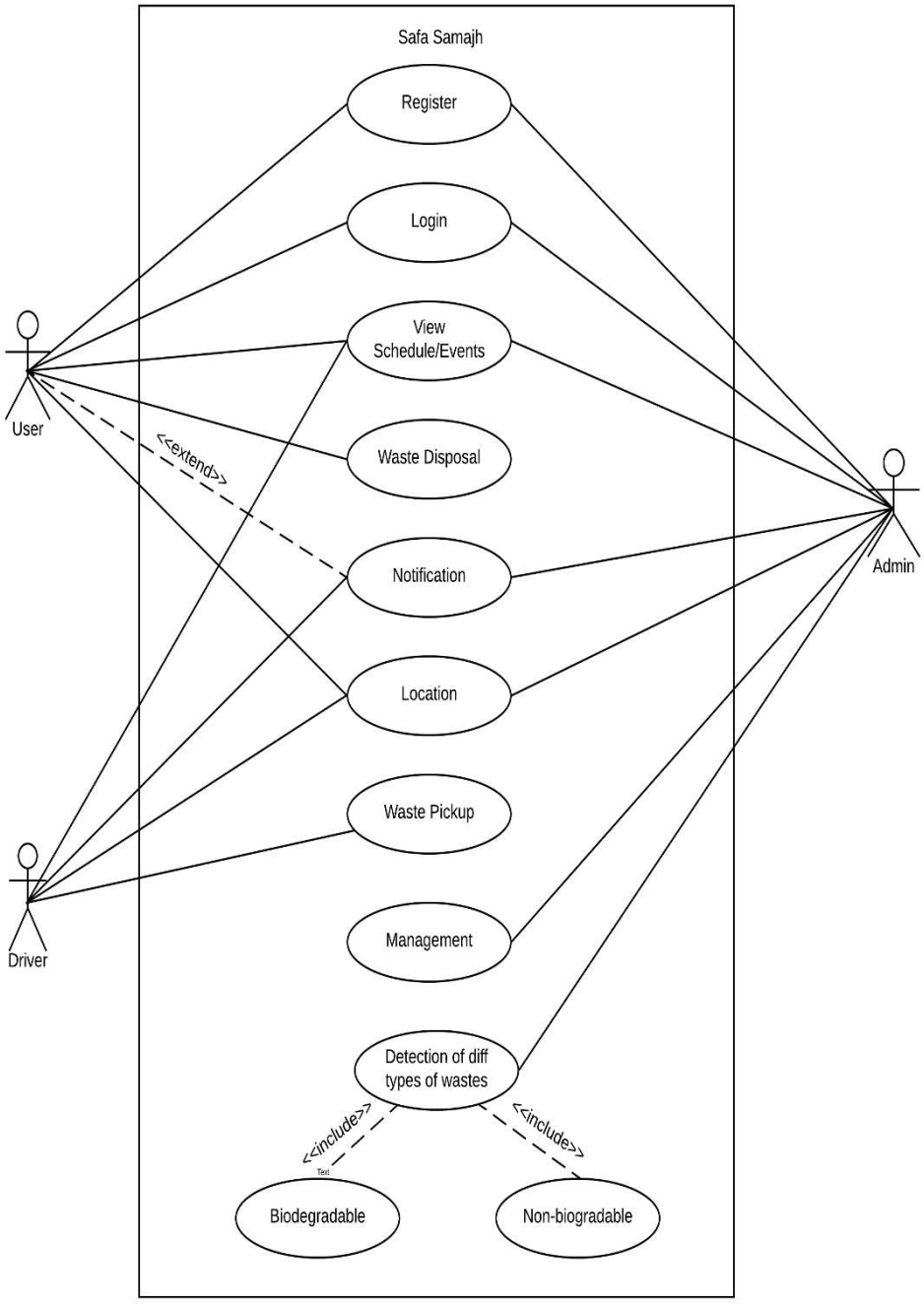


Figure 3.9: Use Case Diagram

3.2.6.2 CLASS DIAGRAM

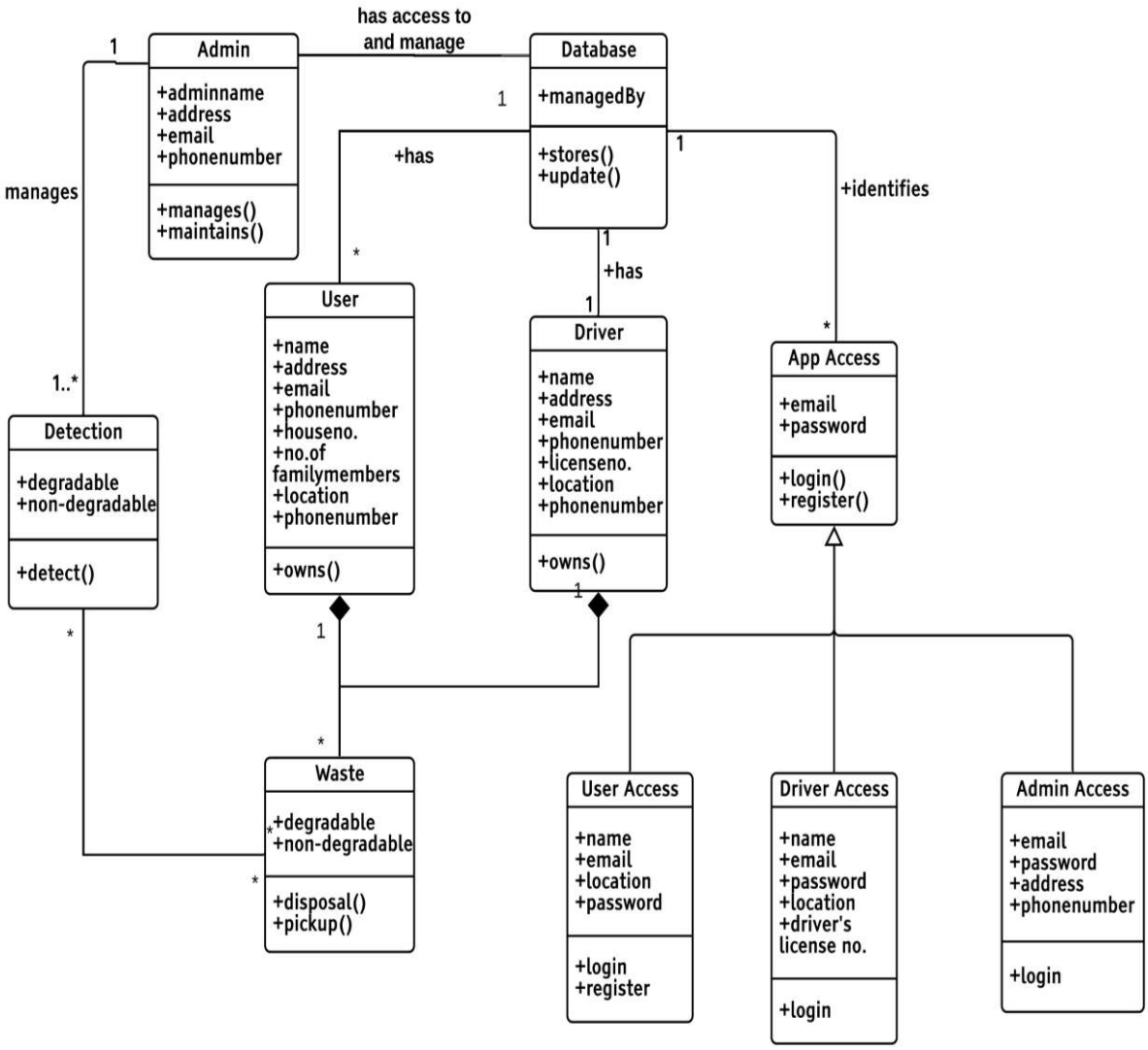


Figure 3.10: Class Diagram

3.2.6.3 SEQUENCE DIAGRAM

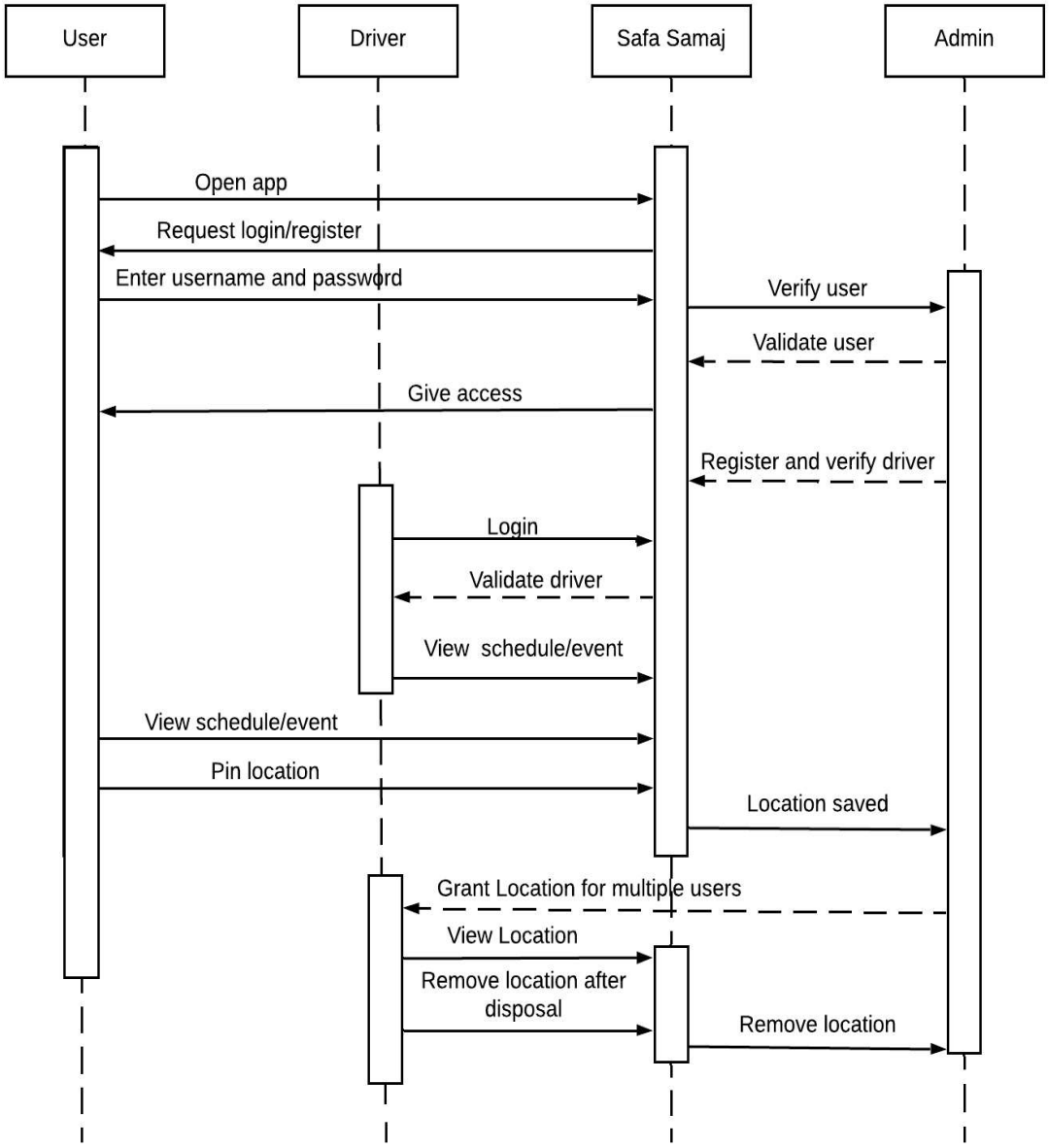


Figure 3.11: Sequence Diagram

CHAPTER 4: RESULT AND ANALYSIS

After completion of this project, two parts were developed. First part was mobile application named “Safa Samaj”. Other part was image classification/image processing using convolutional neural network.

In mobile application part, mobile app was developed to be applicable by user, driver and admin. All these three parties can access to the app. First, they have to undergo registration process. Users can simply register to the app entering details. After registration, users can login to the app. The details of users and the login is saved in admin’s database. We have used firebase as our database. For driver, registration part is done by admin themselves. Driver have to contact the admin and provide their details including license number. This is done in order to authenticate the driver and to avoid any fraud cases. Not everyone could easily open and register their profile as driver unlike users. Driver should be authentic. Admin has the responsibility to manage users and drivers. They can register any drivers, users or even admin. Users can notify driver at the date and time for waste pickup as per their convenience. They have to pin their location. Driver will simply pickup the wastes form the location pinned and take them for management part.

For the image processing part, convolutional neural network is used. This part is used when driver collects the waste from the dumping site and are managed to be segregated on the basis of degradable and non-degradable. The images of wastes collected are captured by external camera. This image is processed and classified as per our trained data. The waste can be processed into categories: glass, paper, plastic, metal, cardboard and degradable.

Due to limitation of hardware (external camera) because of covid crisis, we have implemented the task of external camera using our mobile camera. Images were clicked using mobile camera and sent to the laptop where our model is trained. After passing captured image to the device, the image was processed by our model and it was classified. Result was displayed on the screen along with the probability.

We trained three different data sizes and obtained following results:

First data size includes 4,255 data images resulting

Training accuracy=85.9

Validation accuracy=56.7

Second data size include 2,538 data images resulting

Training accuracy= 79.68

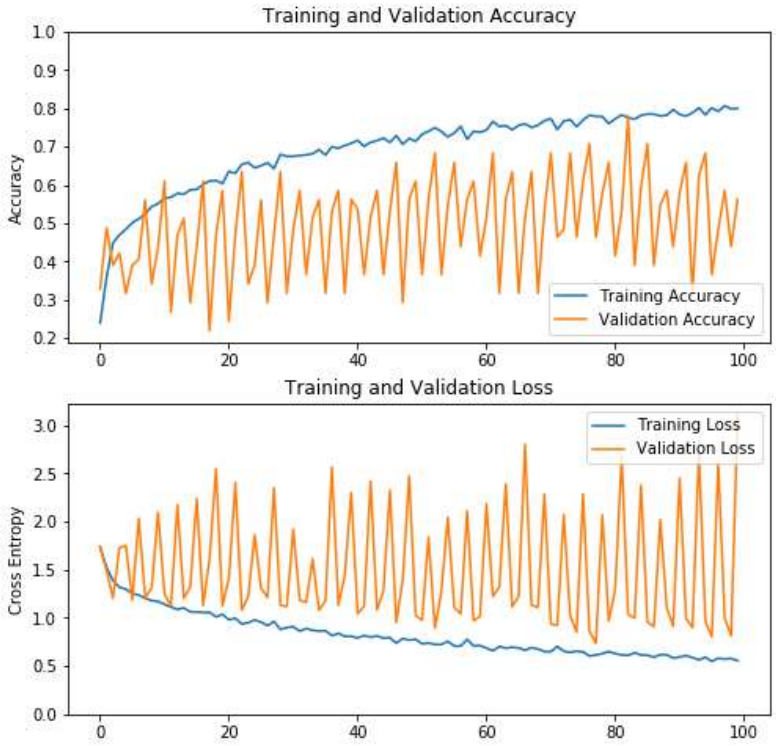
Validation accuracy= 36.78

Third data size include 1,380 data images resulting

Training accuracy=78.7

Validation accuracy= 40.3

For 4255 datasets



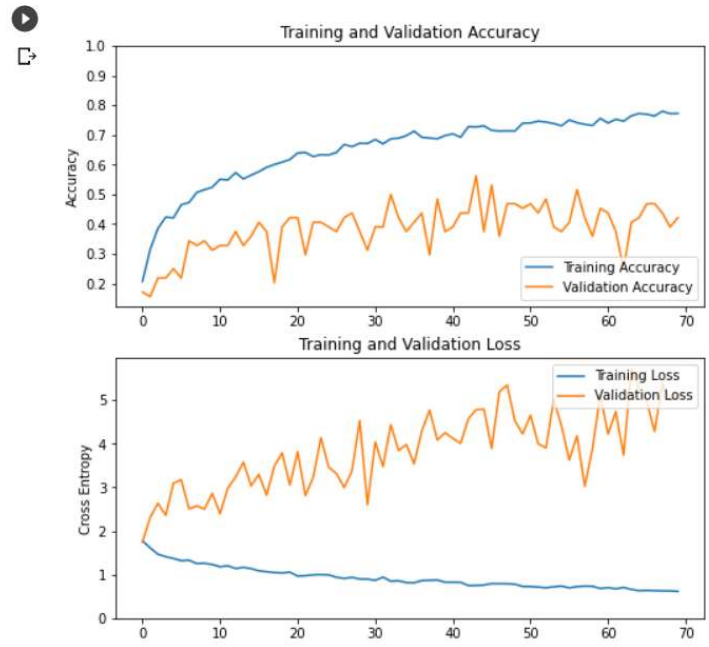
Saving our model

```
In [35]:
1 model.save('Trained_Model_Final.h5')
```

```
In [36]:
1 train_loss, train_acc = model.evaluate_generator(train_generator,steps=16)
2 validation_loss,validation_acc=model.evaluate_generator(validation_generator,steps=16)
3 print('Train: %.3f, Validation: %.3f' %(train_acc,validation_acc))
```

```
Train: 0.859, Validation: 0.567
```

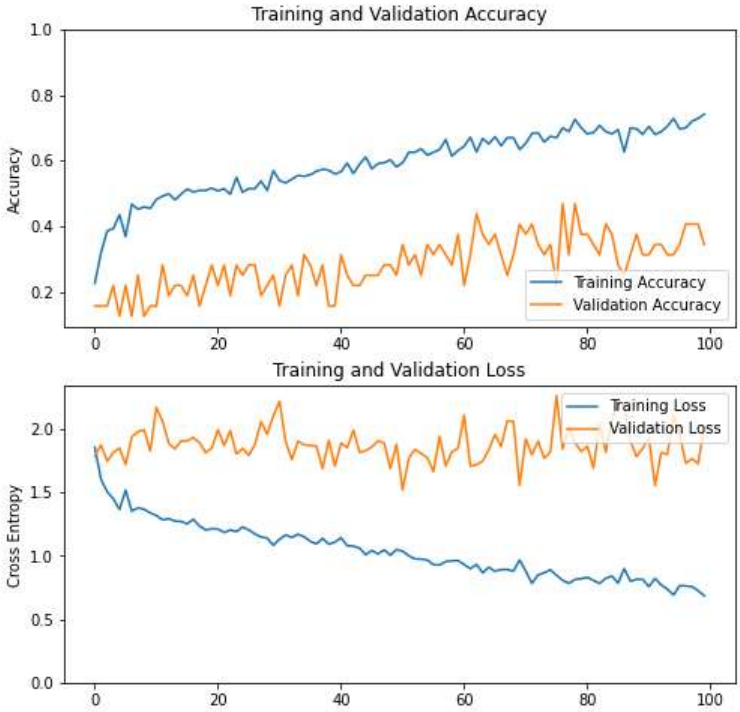
For 2530 datasets




```
train_loss, train_acc=model.evaluate_generator(train_generator,steps=16)
validation_loss,validation_acc=model.evaluate_generator(test_generator,steps=16)
print('Train:%3f, Validation: %3f' %(train_acc,validation_acc))
```

```
/usr/local/lib/python3.7/dist-packages/tensorflow/python/keras/engine/training.py:1877: UserWarning: `Model.evaluate_generator` is deprecat
  warnings.warn('`Model.evaluate_generator` is deprecated and '
WARNING:tensorflow:Your input ran out of data; interrupting training. Make sure that your dataset or generator can generate at least `steps
Train:0.796875, Validation: 0.367816
```

For 1380 datasets




```
[ ] train_loss, train_acc = model.evaluate_generator(train_generator,steps=16)
    validation_loss,validation_acc=model.evaluate_generator(validation_generator,steps=16)
    print('Train: %.3f, Validation: %.3f' %(train_acc,validation_acc))
```

```
/usr/local/lib/python3.6/dist-packages/tensorflow/python/keras/engine/training.py:1877: UserWarning: `Model.evaluate_generator` is deprecated and will be removed
  warnings.warn('`Model.evaluate_generator` is deprecated and '
WARNING:tensorflow:Your input ran out of data; interrupting training. Make sure that your dataset or generator can generate at least `steps_per_epoch * epochs` ba
Train: 0.787, Validation: 0.403
```

CHAPTER 5: VERIFICATION AND VALIDATION

Waste Management System project consists of two parts:

1. App for the user, driver, and admin
2. Waste management by image processing

1. App for User, Driver, and admin:

User, Driver, and Admin can register and login into the app via their email.
Users can register using email and password. After entering into the app they have to enter details: name, address, email, password, house no. and phone no. After this setup user can easily log in to the app thereafter.

When the locality dustbin is full and the driver is to be notified, the user can log in to the app and view the schedule (morning or evening shift). Then they have to go to the location portal. The details of the user are stored at the users' node along with their location stored on the users' location node at the admin's database (firebase). This location detail is passed to the driver location portal.

Not everyone can register themselves as drivers. Driver's account is set by admin. For this the driver has to provide necessary details: name, email, address, password, phone no. and License no. After these details are verified then drivers are allotted and their account is set up by admin. Now drivers can log in to the app and see their schedule. Then have to go to the location portal where the location of multiple users to be covered are available (passed from the database). Drive has to identify the route covering all the user's location.

Admin can setup their account by entering email, password, phone no., address, and admin's name directly at the database (like drivers). They can log in to the app with this email and password to view details regarding the app.

2. Waste Management by image processing

Waste management is the management part after waste from users is collected in a specific area. We have used CNN algorithm for image processing/classification. For this TensorFlow, NumPy and Keras framework is used. The advantage of using the Keras framework is Keras library functions out better in the case of image processing. Large no. of datasets(everyday waste products regarding the paper, glass, plastic, metals, cardboard, and degradable wastes) images are collected and imported into our OpenCV project. Numpy and cv2 are imported for indexing and image loading.

Keras model is used to import convolution model (Conv2D), model checkpoint, early stopping, CSVlogger, and several others. Conv2D provides multiple hidden layers to which our input(image) interacts and single output is obtained as a classified one. EarlyStopping is used to validate accuracy. Map plot function is also used to plot the validation accuracy loss and training accuracy and loss.

For training and validation, we have specified images into two categories train and validation. Sequential Model is used where Image Data Generator generates a class of images for image augmentation. The path of each image set location is passed and all the images are trained. These trained data are saved inside history class. For future use, we save this trained model. Model evaluate generator evaluates all the validation accuracy, loss and train accuracy, loss. This accuracy and loss rate of train and validation are represented in graphical form via the Matplotlib function. After this, we test our prediction by entering any image. These images are classified as per our training model and the probability is also represented in the bar diagram.

After carrying out this training sequential model using CNN we validate that the higher the no. of the dataset better the accuracy and less the loss. We have used 3 different data sizes and results are shown in the result and analysis section above.

CHAPTER 6: CONCLUSION

To conclude, Safa Samaj-Waste Management System can be used as a systematic way for proper waste management in current prevailing situation. Users can notify drivers as per their convenience. Waste pickup dates are also available within less gap. Picked up wastes are also classified into degradable and non-degradable so that they can be used as per reduce, reuse and recycle policy. Hence, total waste pickup and waste management process is systematic as per the project development outline of Safa Samaj. Hence, we concluded:

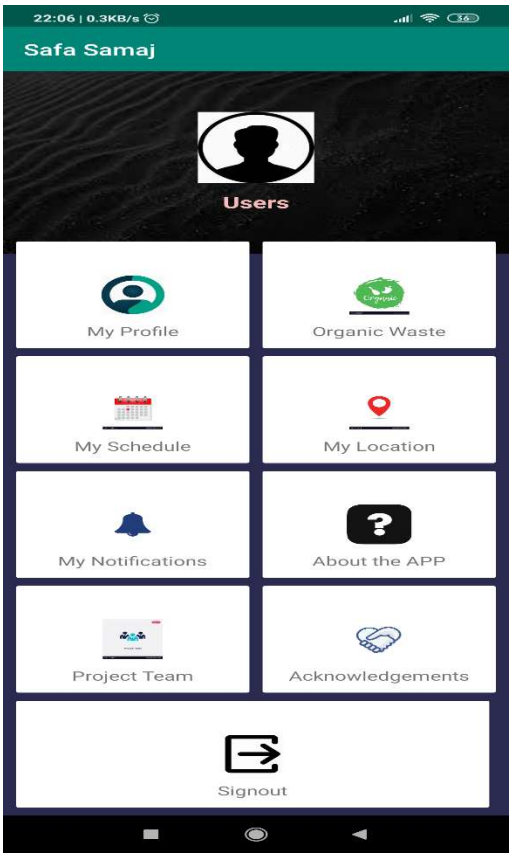
- Analyzation of the current scenario.
- Complete login and register for user drivers and admin.
- Storage of all details of users and drivers in admin's database.
- Locating own location by user and display of all the location of the customers specified to the driver on their map panel.
- Image processing to identify types of wastes.

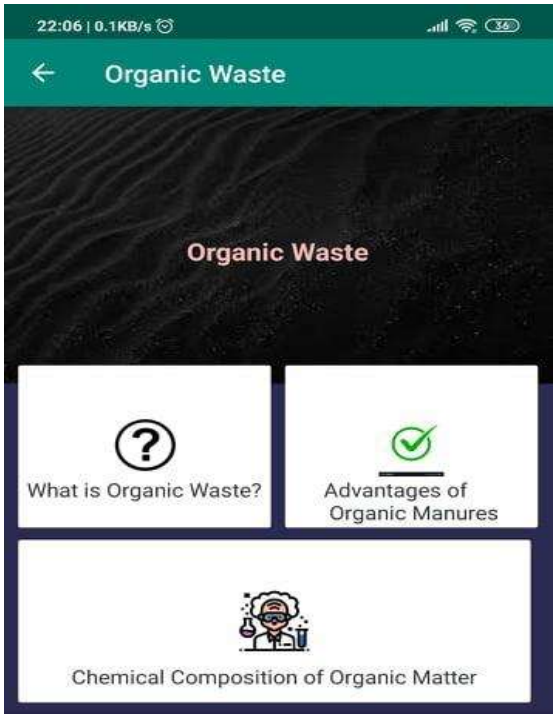
There is still room for some improvements. We aim to implement the following enhancements in future:

- Hardware implementation using external camera for image processing.

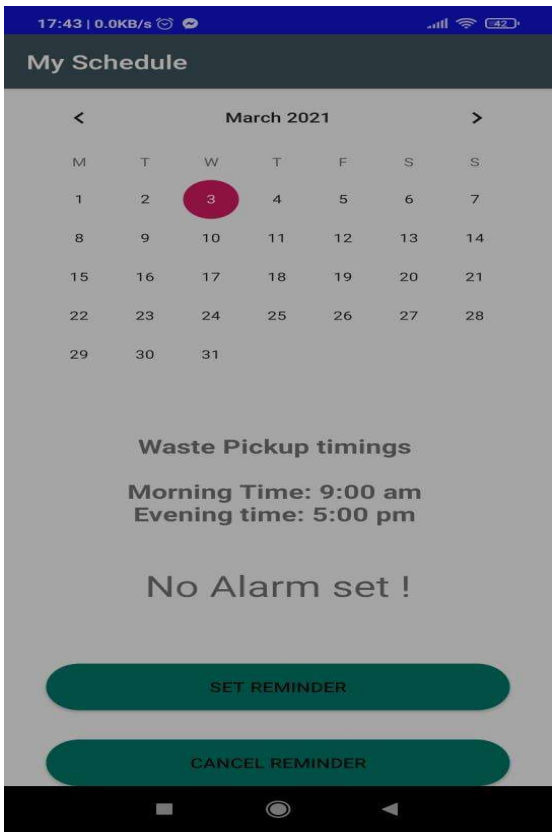
APPENDIX (SCREENSHOTS)

APPLICATION

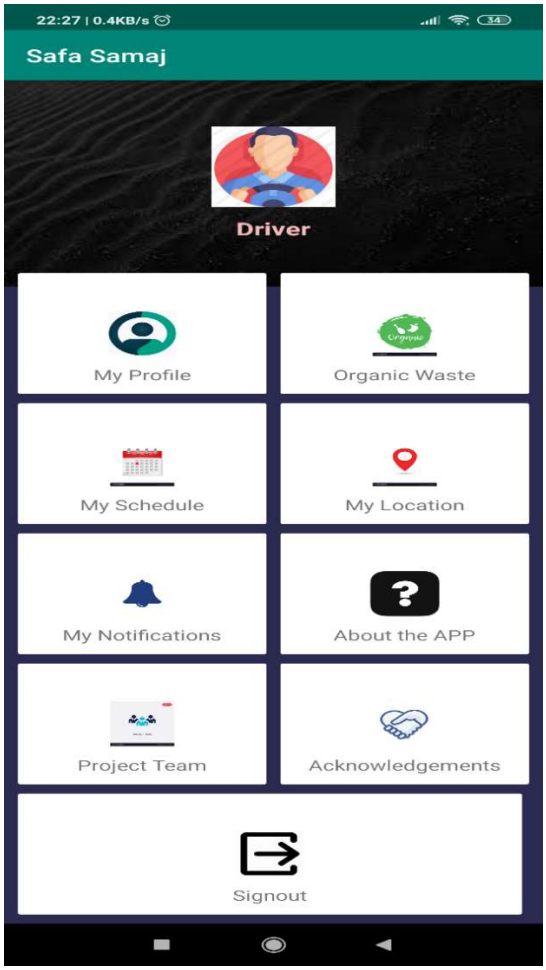


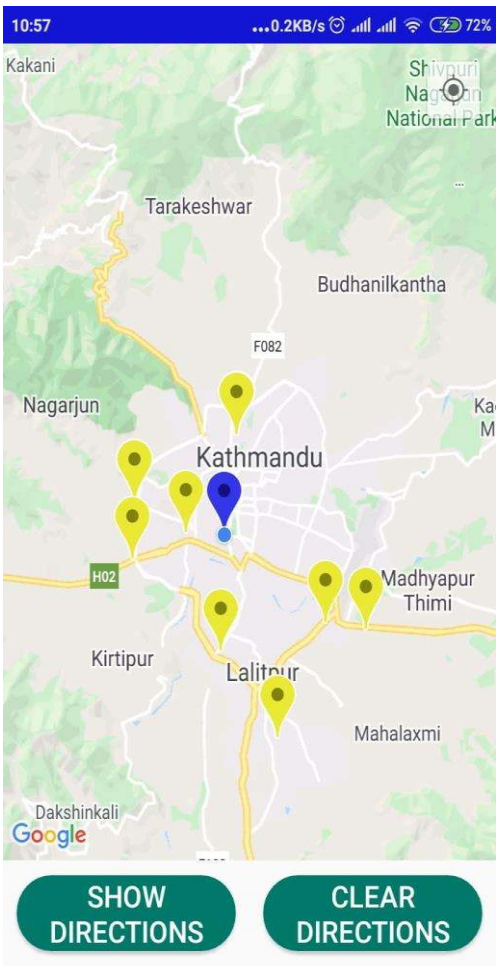

22:06 | 0.1KB/s
Organic Waste
Organic Waste
What is Organic Waste?
Advantages of Organic Manures
Chemical Composition of Organic Matter

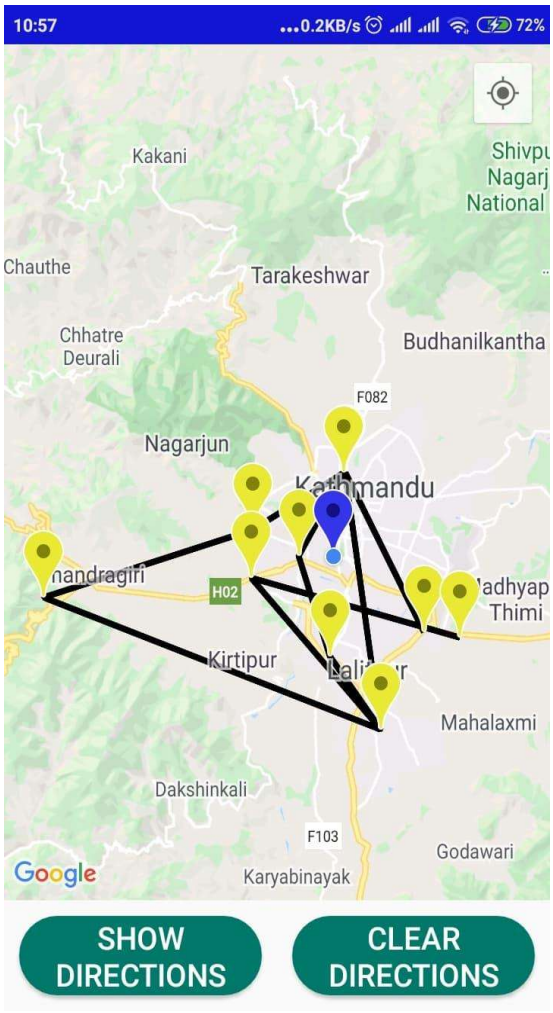

22:07 | 303KB/s
Chaughada
Bhadratar
Madanpur
Sunkhani
Belkot
Kakani
Shivpuri Nagarjun National Park
Okharpauwa
Budhanilkantha
Jeewanpur
Nagarjun
Kathmandu
Kageshwari Manohara
Chandragiri
Kirtipur
Bhaktapur
Chitlang
Dakshinkali
Godawari
Ryale
Karyabinayak
Kulekhani
Dhusel
Nallu
Bhardev
Malta
Sankhu
Jhor Mahankal
Chandanpur
Thingan
Baluwapatti Deupur
Milche
Gundu
Shikharpur
Google

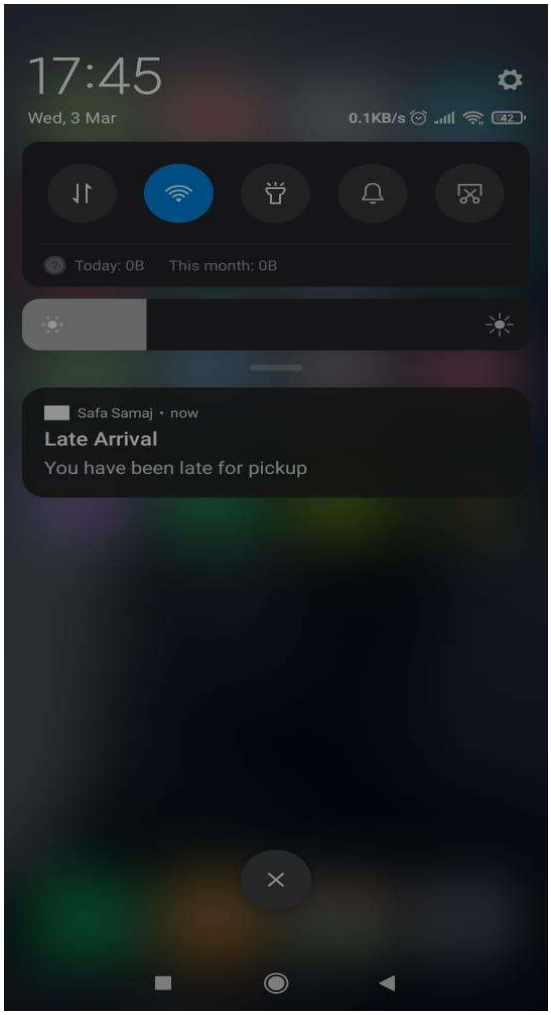

17:43 | 0.0KB/s
My Schedule
March 2021
M T W T F S S
1 2 3 4 5 6 7
8 9 10 11 12 13 14
15 16 17 18 19 20 21
22 23 24 25 26 27 28
29 30 31
Waste Pickup timings
Morning Time: 9:00 am
Evening time: 5:00 pm
No Alarm set !
SET REMINDER
CANCEL REMINDER

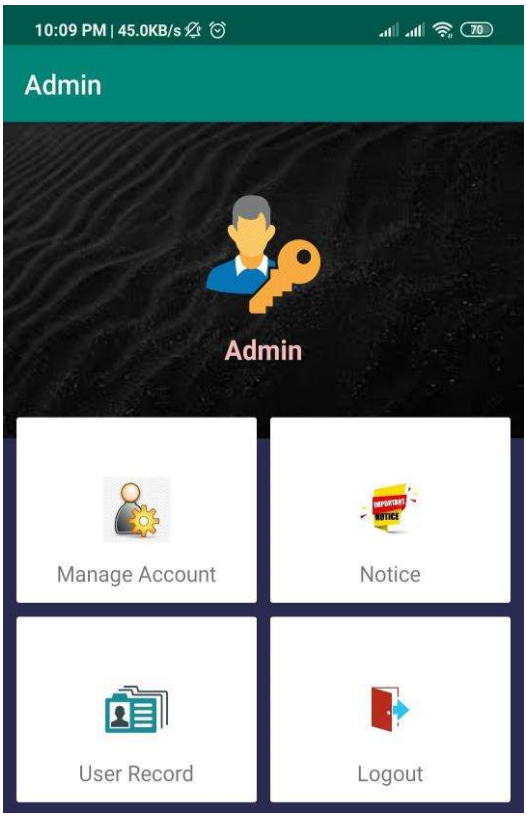

17:43 | 0.8KB/s
Notifications
SEND NOTIFICATION

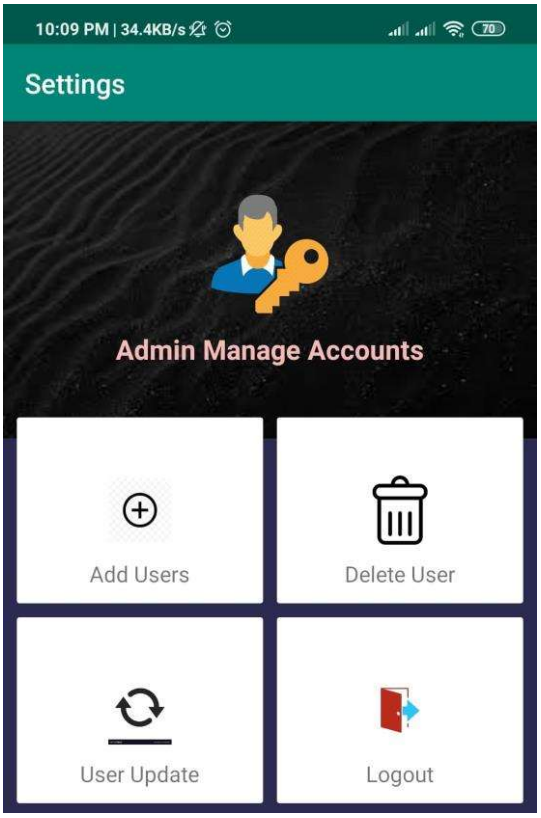

22:27 | 0.4KB/s
Safa Samaj
Driver
My Profile
Organic Waste
My Schedule
My Location
My Notifications
About the APP
Project Team
Acknowledgements
Signout


10:57
0.2KB/s
72%
Kakani
Tarakeshwar
Budhanilkantha
F082
Nagarjun
Kathmandu
H02
Madhyapur
Thimi
Kirtipur
Mahalaxmi
Dakshinkali
Google
SHOW DIRECTIONS
CLEAR DIRECTIONS


10:57
...0.2KB/s
72%
Kakani
Shivpu
Nagarj
National
Chauthe
Tarakeshwar
Chhatre
Deurali
Budhanilkantha
F082
Nagarjun
Kathmandu
H02
Thimi
Kirtipur
Mahalaxmi
Dakshinkali
F103
Godawari
Karyabinayak
Google
SHOW DIRECTIONS
CLEAR DIRECTIONS


17:45
Wed, 3 Mar
0.1KB/s
Today: 0B
This month: 0B
Safa Samaj • now
Late Arrival
You have been late for pickup


10:09 PM | 45.0KB/s
Admin
Admin
Manage Account
Notice
User Record
Logout


10:09 PM | 34.4KB/s
Settings
Admin Manage Accounts
Add Users
Delete User
User Update
Logout

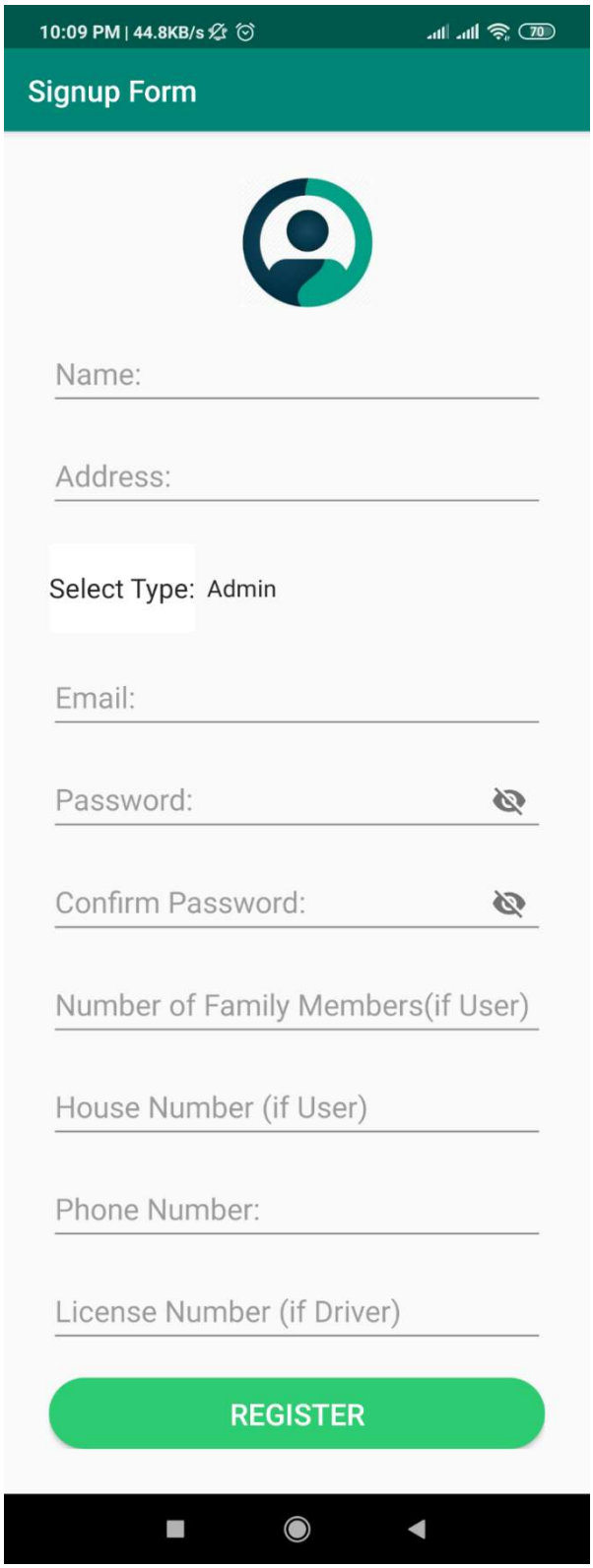

10:09 PM | 44.8KB/s
Signup Form
Name:
Address:
Select Type: Admin
Email:
Password:
Confirm Password:
Number of Family Members(if User)
House Number (if User)
Phone Number:
License Number (if Driver)
REGISTER

IMAGE PROCESSING USING CNN

Image Augmentation

In [21]:
```
train=ImageDataGenerator(horizontal_flip=True,
                         vertical_flip=True,
                         validation_split=0.1,
                         rotation_range= 45,
                         rescale=1./255,
                         shear_range = 0.1,
                         zoom_range = 0.1,
                         width_shift_range = 0.1,
                         height_shift_range = 0.1,)

validation=ImageDataGenerator(rescale=1/255,
                        validation_split=0.1)

train_generator=train.flow_from_directory(train_path,
                                          target_size=(300,300),
                                          batch_size=32,
                                          shuffle=True,
                                          class_mode='categorical',
                                          subset='training')

validation_generator=validation.flow_from_directory(validation_path,
                                        target_size=(300,300),
                                        batch_size=32,
                                        shuffle=False,
                                        class_mode='categorical',
                                        subset='validation')

labels = (train_generator.class_indices)
print(labels)

labels = dict((v,k) for k,v in labels.items())
print(labels)
```

```
Found 3150 images belonging to 6 classes.
Found 73 images belonging to 6 classes.
{'cardboard': 0, 'degradable': 1, 'glass': 2, 'metal': 3, 'paper': 4, 'plastic': 5}
{0: 'cardboard', 1: 'degradable', 2: 'glass', 3: 'metal', 4: 'paper', 5: 'plastic'}
```

Building CNN with Keras

In [30]:
```
model=Sequential()
#Convolution blocks

model.add(Conv2D(32,(3,3), padding='same',input_shape=(300,300,3),activation='relu'))
model.add(MaxPooling2D(pool_size=2))
#model.add(SpatialDropout2D(0.5)) # No accuracy

model.add(Conv2D(64,(3,3), padding='same',activation='relu'))
model.add(MaxPooling2D(pool_size=2))
#model.add(SpatialDropout2D(0.5))

model.add(Conv2D(32,(3,3), padding='same',activation='relu'))
model.add(MaxPooling2D(pool_size=2))

#Classification layers
model.add(Flatten())

model.add(Dense(64,activation='relu'))
#model.add(SpatialDropout2D(0.5))
model.add(Dropout(0.2))
model.add(Dense(32,activation='relu'))

model.add(Dropout(0.2))
model.add(Dense(6,activation='softmax'))
```

Using Logger and callbacks to store the instance of the models

In [31]:
```

filepath="CheckPoints.h5"

checkpoint1 = ModelCheckpoint(filepath, monitor='val_acc', verbose=1, save_best_only=True, mode='max')

#es=EarlyStopping(monitor='val_loss', mode='min', verbose=1, patience=50)

log_csv= CSVLogger('my_logs.csv',separator=',',append=False)

callbacks_list = [checkpoint1,log_csv]
```

Summarizing our models

In [32]:
```
model.summary()
```

```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_4 (Conv2D)            (None, 300, 300, 32)      896
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 150, 150, 32)      0
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 150, 150, 64)      18496
_________________________________________________________________
max_pooling2d_5 (MaxPooling2 (None, 75, 75, 64)        0
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 75, 75, 32)        18464
_________________________________________________________________
max_pooling2d_6 (MaxPooling2 (None, 37, 37, 32)        0
_________________________________________________________________
flatten_2 (Flatten)          (None, 43808)             0
_________________________________________________________________
dense_4 (Dense)              (None, 64)                2803776
_________________________________________________________________
dropout_3 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_5 (Dense)              (None, 32)                2080
_________________________________________________________________
dropout_4 (Dropout)          (None, 32)                0
_________________________________________________________________
dense_6 (Dense)              (None, 6)                 198
=================================================================
Total params: 2,843,910
Trainable params: 2,843,910
Non-trainable params: 0
```

Traning our model

In [34]:
```
history = model.fit_generator(train_generator,
                              epochs=100,
                              steps_per_epoch=3150//32,
                              validation_data=validation_generator,
                              validation_steps=73//32,
                              workers = 4,
                              callbacks=callbacks_list)
#41 epoch - 75% #73- 76.9%
#78 epoch - 80%
```

```
Epoch 1/100
98/98 [==============================] - 465s 5s/step - loss: 1.7299 - acc: 0.2396 - val_loss: 1.7424 - val_acc: 0.3281

Epoch 00001: val_acc improved from -inf to 0.32812, saving model to CheckPoints.h5
Epoch 2/100
98/98 [==============================] - 456s 5s/step - loss: 1.5249 - acc: 0.3586 - val_loss: 1.4904 - val_acc: 0.4878

Epoch 00002: val_acc improved from 0.32812 to 0.48780, saving model to CheckPoints.h5
Epoch 3/100
98/98 [==============================] - 430s 4s/step - loss: 1.3896 - acc: 0.4484 - val_loss: 1.2048 - val_acc: 0.3902

Epoch 00003: val_acc did not improve from 0.48780
Epoch 4/100
98/98 [==============================] - 414s 4s/step - loss: 1.3182 - acc: 0.4699 - val_loss: 1.7278 - val_acc: 0.4219

Epoch 00004: val_acc did not improve from 0.48780
Epoch 5/100
98/98 [==============================] - 360s 4s/step - loss: 1.2983 - acc: 0.4840 - val_loss: 1.7514 - val_acc: 0.3171
```

Saving our model

In [35]:
```
model.save('Trained_Model_Final.h5')
```

In [36]:
```
train_loss, train_acc = model.evaluate_generator(train_generator,steps=16)
validation_loss,validation_acc=model.evaluate_generator(validation_generator,steps=16)
print('Train: %.3f, Validation: %.3f' %(train_acc,validation_acc))
```

```
Train: 0.859, Validation: 0.567
```

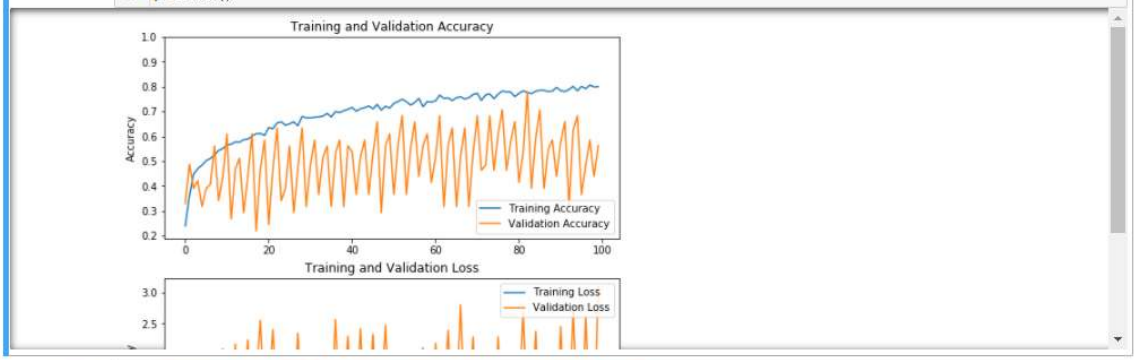
Plotting the Datas:

Accuracy Graphs

In [37]:

```
acc = history.history['acc']
val_acc = history.history['val_acc']

loss = history.history['loss']
val_loss = history.history['val_loss']

# _________________ Graph 1 -------------------------

plt.figure(figsize=(8, 8))
plt.subplot(2, 1, 1)
plt.plot(acc, label='Training Accuracy')
plt.plot(val_acc, label='Validation Accuracy')
plt.legend(loc='lower right')
plt.ylabel('Accuracy')
plt.ylim([min(plt.ylim()),1])
plt.title('Training and Validation Accuracy')

# _________________ Graph 2 -------------------------

plt.subplot(2, 1, 2)
plt.plot(loss, label='Training Loss')
plt.plot(val_loss, label='Validation Loss')
plt.legend(loc='upper right')
plt.ylabel('Cross Entropy')
plt.ylim([0,max(plt.ylim())])
plt.title('Training and Validation Loss')
plt.show()
```

```
plt.plot(loss, label='Training Loss')
plt.plot(val_loss, label='Validation Loss')
plt.legend(loc='upper right')
plt.ylabel('Cross Entropy')
plt.ylim([0,max(plt.ylim())])
plt.title('Training and Validation Loss')
plt.show()
```



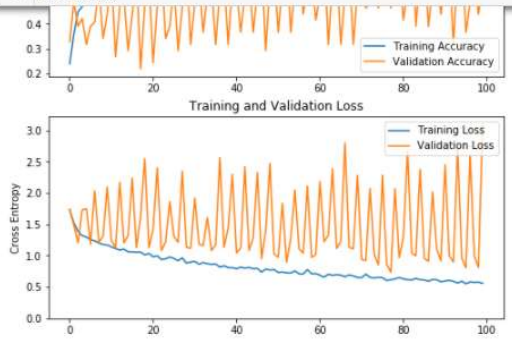
In [ ]:

```
#print("Values stored in the hisotry are ....\n",history.history)
#plt.plot(history.history['loss'],label='train')
#plt.plot(history.history['val_loss'],label='validation')
#plt.legend()
#plt.show()
```

In [ ]:

```
#print("Values stored in the hisotry are ....\n",history.history)
#plt.plot(history.history['acc'],label='train')
```

```
plt.plot(loss, label='Training Loss')
plt.plot(val_loss, label='Validation Loss')
plt.legend(loc='upper right')
plt.ylabel('Cross Entropy')
plt.ylim([0,max(plt.ylim())])
plt.title('Training and Validation Loss')
plt.show()
```



In [ ]:
```
#print("Values stored in the hisotry are ....\n",history.history)
#plt.plot(history.history['loss'],label='train')
#plt.plot(history.history['val_loss'],label='validation')
#plt.legend()
#plt.show()
```

In [ ]:
```
#print("Values stored in the hisotry are ....\n",history.history)
#plt.plot(history.history['acc'],label='train')
```

Saving our model after traning.

In [142]:
```
model.save('Trained_Model.h5')
```

In [143]:
```
history.history
```

```
0.49179205,
0.49042407,
0.50410396,
0.54347825,
0.5344353,
0.54211956,
0.55509645,
0.54309165,
0.546875,
0.5736915,
0.5683994,
0.5745554,
0.5506156,
0.57387143,
0.5827633,
0.59028727,
0.6148098,
0.5736915,
0.5808424,
0.60328317,
```

In [156]:
```
train_loss, train_acc = model.evaluate_generator(train_generator,steps=16)
validation_loss,validation_acc=model.evaluate_generator(validation_generator,steps=16)
print('Train: %.3f, Validation: %.3f' %(train_acc,validation_acc))
```

```
Train: 0.725, Validation: 0.516
```

Testing our Prediction

In [47]:

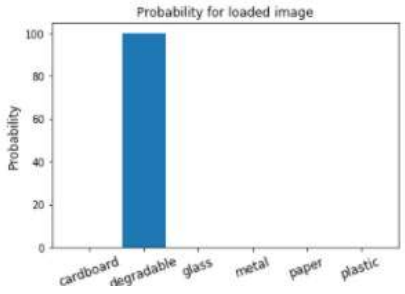
```
from keras.preprocessing import image

img_path = 'D:/Lasttry/validation/degradable/degradable (2).jpg'

img = image.load_img(img_path, target_size=(300, 300))
img = image.img_to_array(img, dtype=np.uint8)
img=np.array(img)/255.0

plt.title("Loaded Image")
plt.axis('off')
plt.imshow(img.squeeze())

p=model.predict(img[np.newaxis, ...])

#print("Predicted shape",p.shape)
print("Maximum Probability: ",np.max(p[0], axis=-1))
predicted_class = labels[np.argmax(p[0], axis=-1)]
print("Classified:",predicted_class)
```

```
Maximum Probability:  0.99995685
Classified: degradable
```

In [48]:

```
classes=[]
prob=[]
print("\n-------------------Individual Probability--------------------------------\n")

for i,j in enumerate (p[0],0):
    print(labels[i].upper(),':',round(j*100,2),'%')
    classes.append(labels[i])
    prob.append(round(j*100,2))

def plot_bar_x():
    # this is for plotting purpose
    index = np.arange(len(classes))
    plt.bar(index, prob)
    plt.xlabel('Labels', fontsize=12)
    plt.ylabel('Probability', fontsize=12)
    plt.xticks(index, classes, fontsize=12, rotation=20)
    plt.title('Probability for loaded image')
    plt.show()
plot_bar_x()
```

```
-------------------Individual Probability--------------------------------

CARDBOARD : 0.0 %
DEGRADABLE : 100.0 %
GLASS : 0.0 %
METAL : 0.0 %
PAPER : 0.0 %
PLASTIC : 0.0 %
```



Testing Prediction.

In [175]:

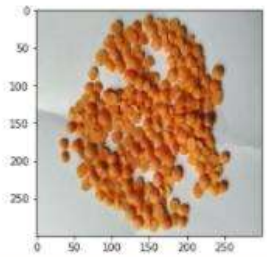
```
from keras.preprocessing import image

img_path = 'D:/UdacityTensorflow/MajorProject/DataSets/validation/cardboard/cardboard (15).jpg'

img = image.load_img(img_path, target_size=(300, 300))
img = image.img_to_array(img, dtype=np.uint8)
img=np.array(img)/255.0

plt.title("Loaded Image")
plt.axis('off')
plt.imshow(img.squeeze())

p=model.predict(img[np.newaxis, ...])

#print("Predicted shape",p.shape)
print("Maximum Probability: ",np.max(p[0], axis=-1))
predicted_class = labels[np.argmax(p[0], axis=-1)]
print("Classified:",predicted_class)

```

```
Maximum Probability:  0.9679967
Classified: cardboard
```

In [5]:

```

def predict(img_path):
    labels={0: 'cardboard', 1: 'degradable', 2: 'glass', 3: 'metal', 4: 'paper', 5: 'plastic', 6: 'trash'}
    img_path = 'D:/Lasttry/validation/(50).jpg'

    img = image.load_img(img_path, target_size=(300, 300))
    img = image.img_to_array(img, dtype=np.uint8)
    img=np.array(img)/255.0
    plt.imshow(img.squeeze())

    model = tf.keras.models.load_model("Trained_Model.h5")
    p=model.predict(img[np.newaxis, ...])
    pro=np.max(p[0], axis=-1)
    print("p.shape:",p.shape)
    print("prob",pro)
    predicted_class = labels[np.argmax(p[0], axis=-1)]
    #os.remove(img_path)
    print("classified label:",predicted_class)
    return(str(predicted_class)+" \n Probability:"+str(pro))
print(predict(img_path = 'D:/Lasttry/validation/degradable/degradable (50).jpg'))
```

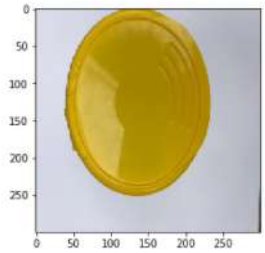
```
p.shape: (1, 7)
prob 0.97586036
classified label: degradable
degradable
 Probability:0.97586036
```



In [11]:

```
def predict(img_path):
     labels={0: 'cardboard', 1: 'degradable', 2: 'glass', 3: 'metal', 4: 'paper', 5: 'plastic'}
     img_path = 'D:/Lasttry/validation/plastic/plastic (145).jpg'

     img = image.load_img(img_path, target_size=(300, 300))
     img = image.img_to_array(img, dtype=np.uint8)
     img=np.array(img)/255.0
     plt.imshow(img.squeeze())

     model = tf.keras.models.load_model("Trained_Model_Final.h5")
     p=model.predict(img[np.newaxis, ...])
     pro=np.max(p[0], axis=-1)
     print("p.shape:",p.shape)
     print("prob",pro)
     predicted_class = labels[np.argmax(p[0], axis=-1)]
     #os.remove(img_path)
     print("classified label:",predicted_class)
     return(str(predicted_class)+" \n Probability:"+str(pro))
print(predict(img_path = 'D:/Lasttry/validation/plastic/plastic (145).jpg'))
```

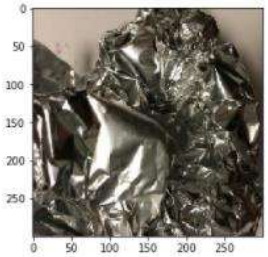
```
p.shape: (1, 6)
prob 0.9846028
classified label: plastic
plastic
 Probability:0.9846028
```



In [16]:

```
def predict(img_path):
     labels={0: 'cardboard', 1: 'degradable', 2: 'glass', 3: 'metal', 4: 'paper', 5: 'plastic'}
     img_path = 'D:/Lasttry/validation/metal/metal (36).jpg'

     img = image.load_img(img_path, target_size=(300, 300))
     img = image.img_to_array(img, dtype=np.uint8)
     img=np.array(img)/255.0
     plt.imshow(img.squeeze())

     model = tf.keras.models.load_model("Trained_Model_Final.h5")
     p=model.predict(img[np.newaxis, ...])
     pro=np.max(p[0], axis=-1)
     print("p.shape:",p.shape)
     print("prob",pro)
     predicted_class = labels[np.argmax(p[0], axis=-1)]
     #os.remove(img_path)
     print("classified label:",predicted_class)
     return(str(predicted_class)+" \n Probability:"+str(pro))
print(predict(img_path = 'D:/Lasttry/validation/metal/metal (36).jpg'))
```

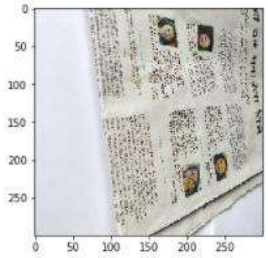
```
p.shape: (1, 6)
prob 0.9977701
classified label: metal
metal
 Probability:0.9977701
```



In [12]:

```
def predict(img_path):
     labels={0: 'cardboard', 1: 'degradable', 2: 'glass', 3: 'metal', 4: 'paper', 5: 'plastic'}
     img_path = 'D:/Lasttry/validation/paper/paper (15).jpg'

     img = image.load_img(img_path, target_size=(300, 300))
     img = image.img_to_array(img, dtype=np.uint8)
     img=np.array(img)/255.0
     plt.imshow(img.squeeze())

     model = tf.keras.models.load_model("Trained_Model_Final.h5")
     p=model.predict(img[np.newaxis, ...])
     pro=np.max(p[0], axis=-1)
     print("p.shape:",p.shape)
     print("prob",pro)
     predicted_class = labels[np.argmax(p[0], axis=-1)]
     #os.remove(img_path)
     print("classified label:",predicted_class)
     return(str(predicted_class)+" \n Probability:"+str(pro))
print(predict(img_path = 'D:/Lasttry/validation/paper/paper (15).jpg'))
```

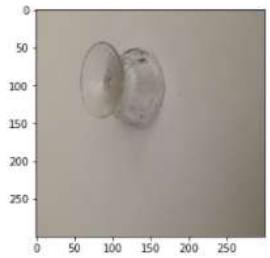
```
p.shape: (1, 6)
prob 0.5707775
classified label: paper
paper
 Probability:0.5707775
```



In [17]:

```
def predict(img_path):
     labels={0: 'cardboard', 1: 'degradable', 2: 'glass', 3: 'metal', 4: 'paper', 5: 'plastic'}
     img_path = 'D:/Lasttry/validation/glass/glass (36).jpg'

     img = image.load_img(img_path, target_size=(300, 300))
     img = image.img_to_array(img, dtype=np.uint8)
     img=np.array(img)/255.0
     plt.imshow(img.squeeze())

     model = tf.keras.models.load_model("Trained_Model_Final.h5")
     p=model.predict(img[np.newaxis, ...])
     pro=np.max(p[0], axis=-1)
     print("p.shape:",p.shape)
     print("prob",pro)
     predicted_class = labels[np.argmax(p[0], axis=-1)]
     #os.remove(img_path)
     print("classified label:",predicted_class)
     return(str(predicted_class)+" \n Probability:"+str(pro))
print(predict(img_path = 'D:/Lasttry/validation/glass/glass (36).jpg'))
```

```
p.shape: (1, 6)
prob 0.72532356
classified label: glass
glass
 Probability:0.72532356
```



REFERENCES

[1] "Solid Waste Management in Nepal: Current Status and Policy Recommendations," Aug 2013. [Online]. Available: https://www.adb.org/publications/solid-waste-management-nepal-current-status-and-policy-recommendations. [Accessed 1 Dec 2019].

[2] "Solid Waste.," United States Environmental Protection Agency., 19 May 2019. [Online]. Available: https://www.epa.gov/hw/final-rule-2018-definition-solid-waste-dswresponse-court-vacatur. . [Accessed 1 Dec 2019].

[3] "prokerala," 23 July 2012. [Online]. Available: https://www.prokerala.com/going-green/solidwastes.htm . [Accessed 1 Dec 2019].

[4] "History of waste management," 1 Nov 2019. [Online]. Available: https://en.wikipedia.org/wiki/History_of_waste_management. [Accessed 1 Dec 2019].

[5] "THE WORLD BANK," 1 April 2019. [Online]. Available: https://www.worldbank.org/en/topic/urbandevelopment/brief/solid-waste-management . [Accessed 1 Dec 2019].

[6] Khalisisi, [Online]. Available: https://khaalisisi.com/. [Accessed 1 Dec 2019].

[7] [Online]. Available: https://dokorecyclers.com/about.

[8] "Iterative Model," 15 Dec 2016. [Online]. Available: https://airbrake.io/blog/sdlc/iterative-model. [Accessed 1 Dec 2019].

[9] A. A. Yuzhakov, "Convolutional Neural Networks Application inPlastic Waste Recognition and Sorting," 01 2018.

[10] Prabhu, "Understanding of Convolutional Neural Network (CNN) — Deep Learning," 2018.

[11] M. Rouse, "Android Studio," 2003. [Online]. Available: https://searchmobilecomputing.techtarget.com/definition/Android-Studio. [Accessed 1 Dec 2019].

[12] "Java (programming language)," [Online]. Available: https://en.wikipedia.org/wiki/Java_(programming_language).

[13] "Firebase," [Online]. Available: https://en.wikipedia.org/wiki/Firebase. [Accessed 1 Dec 2019].

[14] "Keras: Deep Learning library for Theano and TensorFlow".

[15] "What is TensorFlow? The machine learning library explained," 18 07 1998.