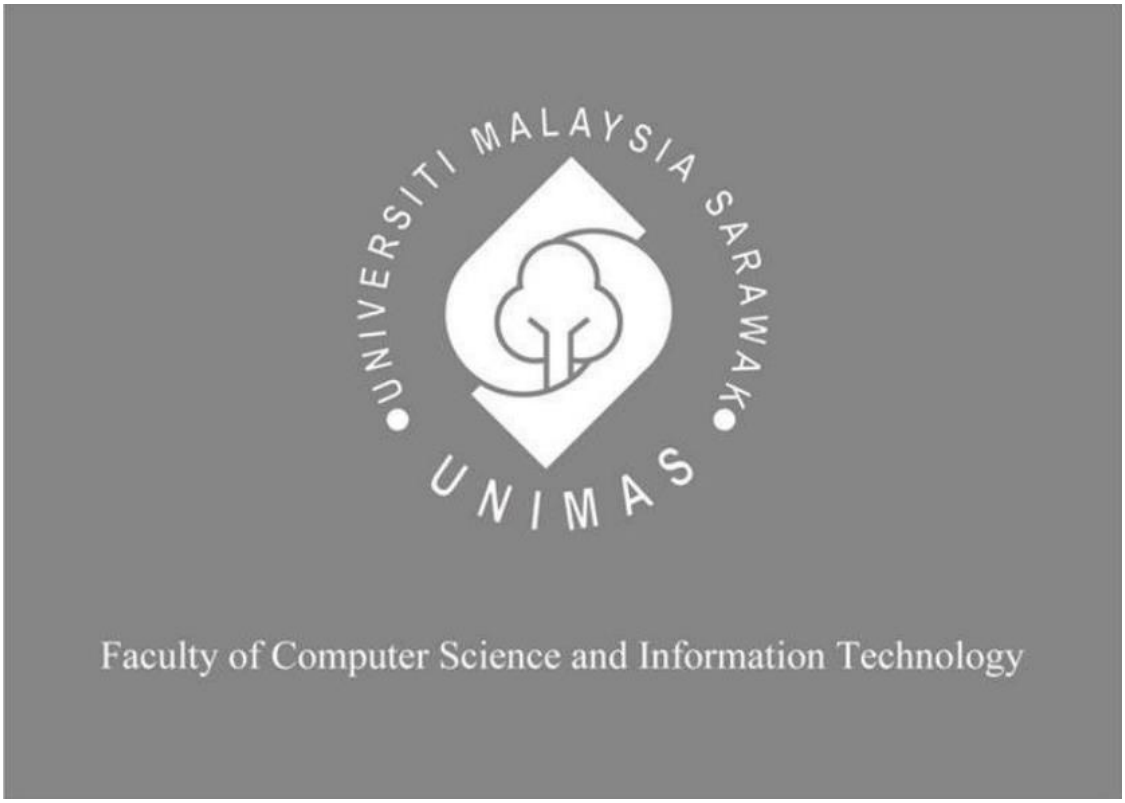


WASTE SORTING APP WITH IMAGE RECOGNITION

Nur Hafizah Binti Ramli

Bachelor of Software Engineering with Honours

2025

WASTE SORTING APP WITH IMAGE RECOGNITION

NUR HAFIZAH BINTI RAMLI

80609

This project is submitted in partial fulfilment of

requirements for the degree of

Bachelor of Software Engineering with Honours

Faculty of Computer Science and Information Technology

UNIVERSITI MALAYSIA SARAWAK

2025

MENTOR PINTAR DENGAN PENGAWAL SUARA

NUR HAFIZAH BINTI RAMLI

80609

Project ini merupakan salah satu keperluan untuk

Ijazah Sarjana Muda Kejuruteraan Perisian dengan Kepujian

Fakulti Sains Komputer dan Teknologi Maklumat

UNIVERSITI MALAYSIA SARAWAK

2025

UNIVERSITI MALAYSIA SARAWAK

THESIS STATUS ENDORSEMENT FORM

TITLE WASTE SORTING APP WITH IMAGE RECOGNITION

ACADEMIC SESSION: 2024/2025

NUR HAFIZAH BINTI RAMLI

(CAPITAL LETTERS)

hereby agree that this Thesis* shall be kept at the Centre for Academic Information Services, Universiti Malaysia Sarawak, subject to the following terms and conditions:

1. The Thesis is solely owned by Universiti Malaysia Sarawak
2. The Centre for Academic Information Services is given full rights to produce copies for educational purposes only
3. The Centre for Academic Information Services is given full rights to do digitization in order to develop local content database
4. The Centre for Academic Information Services is given full rights to produce copies of this Thesis as part of its exchange item program between Higher Learning Institutions [or for the purpose of interlibrary loan between HLI]
5. ** Please tick (√)

☐	CONFIDENTIAL	(Contains classified information bounded by the OFFICIAL SECRETS ACT 1972)
☐	RESTRICTED	(Contains restricted information as dictated by the body or organization where the research was conducted)
☑	UNRESTRICTED	

Validated by

(AUTHOR'S SIGNATURE)

(SUPERVISOR'S SIGNATURE)

Permanent Address

JALAN JASMIN KDB 1,
96000 SIBU,
SARAWAK

Date: 17/01/25

Date: _____

Note * Thesis refers to PhD, Master, and Bachelor Degree
** For Confidential or Restricted materials, please attach relevant documents from relevant organizations / authorities

DECLARATION

I hereby declare that the project is my original work. I have not copied from any other student's work or from any other sources except due to reference or acknowledgment is not made explicitly in the text, nor has any part had been written for me by another person.

(NUR HAFIZAH BINTI RAMLI, 80609)

ACKNOWLEDGMENT

I would like to take this opportunity to say thank you to everyone who helped and guided me throughout this final year project. First and foremost, my sincere thanks go to my supervisor, Associate Professor Ts. Dr. Hamimah binti Ujir. She provided me with insightful feedback and encouraged me to think critically about my work. Her help has been very important to me. Besides, she also provides ideas on how to plan, design, develop and enhance this project. Without her support, the progression of the project could not go so smoothly.

Next, I want to express my sincere gratitude to the FYP coordinator, Professor Dr. Wang Yin Chai for his guidance, and continuous feedback throughout the FYP period. I would also like to thank my examiner, Associate Professor Dr. Jacey Lynn Minoi for taking the time to review my project. I would like to say thank you to my friends and classmates who have given encouragement and support to me and share related information with me. Finally, I want to express my gratitude to my family for their love and support throughout my studies. Without their support and advice, the completion of the project could not be accomplished. Thank you all.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

As technology advances and awareness of environmental issues grows, individuals increasingly seek innovative solutions to improve waste management and recycling practices. Many people still have a difficulty in properly sorting waste due to limited knowledge and understanding of waste categories. Common items, including batteries, paints, and electronic components that require special disposal methods, yet many individuals are unaware of these requirements. To address these challenges, the proposed application "Waste Sorting App with Image Recognition" is developed to simplify the waste sorting process through a mobile application that utilizes an advanced image recognition system powered by AI algorithms. This application allows users to capture images of various waste items and receive instant feedback on their types to be able to sort them correctly. The AI algorithms analyze the images to identify the type of waste and provide a clear guidance on whether it is recyclable, compostable, hazardous, or general waste. By integrating machine learning techniques, the app improves its accuracy over time and can adapt to user interactions. In contrast with mechanical methods and IoT systems, AI has the potential to process complex data and also a more cost-effective solution. This project adopts an agile development approach, emphasizing iterative progress and continuous feedback, which allows for flexibility throughout the project lifecycle. To align the application with user requirements, a survey was conducted to gather insights on preferences regarding waste sorting. A comparative analysis of similar applications was also performed to identify their limitations to ensure that the proposed application implements valuable features. System architecture and database design are represented through UML diagrams, while the user interface design is created using wireframing methods. This project aims not only to improve waste sorting habits but also to promote a more sustainable lifestyle by making recycling more accessible and efficient through the integration of AI-powered image recognition technology.

ABSTRAK

Seiring dengan kemajuan teknologi dan peningkatan kesedaran tentang isu alam sekitar, setiap individu meningkatkan pencarian dalam menyelesaikan isu tersebut untuk menambah baik pengurusan sisa dan amalan kitar semula. Masih banyak yang menghadapi kesukaran untuk mengasingkan sisa-sisa dengan betul kerana pengetahuan dan pemahaman yang terhad tentang jenisnya. Barangan biasa seperti bateri, cat dan komponen elektronik yang memerlukan kaedah pelupusan khas, namun ramai individu masih tidak mengetahui keperluan ini. Untuk menangani cabaran-cabaran ini, aplikasi yang dicadangkan iaitu "Waste Sorting App with Image Recognition" dibangunkan untuk memudahkan proses pengasingan sisa melalui aplikasi mudah alih yang menggunakan sistem pengecaman imej yang canggih yang dikuasakan oleh algoritma AI. Aplikasi ini membolehkan para pengguna untuk mengambil gambar pelbagai sisa dan menerima maklum balas tentang jenis-jenis sisa tersebut dengan segera supaya dapat mengasingkannya dengan betul. Algoritma AI tersebut menganalisis gambar-gambar untuk mengenal pasti tentang jenis sisa dan memberi panduan yang jelas sama ada ia boleh dikitar semula, kompos, berbahaya atau sisa am. Dengan mengintegrasikan teknik pembelajaran mesin, aplikasi ini meningkatkan ketepatannya dari semasa ke semasa dan dapat menyesuaikan diri dengan interaksi pengguna. AI mempunyai potensi untuk memproses data yang kompleks dan juga merupakan penyelesaian yang lebih kos efektif berbanding dengan kaedah mekanikal dan sistem IoT. Projek ini menggunakan pendekatan pembangunan yang tangkas, menekankan kemajuan secara iteratif dan maklum balas yang berterusan. Hal ini membolehkan fleksibiliti sepanjang kitaran hayat projek. Satu tinjauan telah dijalankan untuk mengumpul pendapat tentang pengasingan sisa untuk menyelaraskan aplikasi dengan keperluan pengguna. Satu analisis untuk membandingkan perbezaan di antara aplikasi-aplikasi yang serupa juga dijalankan untuk mengenal pasti hadnya bagi memastikan aplikasi yang dicadangkan melaksanakan ciri-ciri yang berguna. Reka bentuk seni bina sistem dan pangkalan data diwakili melalui gambar rajah UML, manakala halaman grafik aplikasi direka menggunakan kaedah wireframe. Projek ini bertujuan bukan sahaja untuk memperbaiki tabiat pengasingan sisa tetapi juga untuk mempromosikan gaya hidup yang lebih lestari dalam menjadikan kitar semula lebih mudah diakses dan cekap melalui pengintegrasian teknologi pengenalan imej yang dikuasakan oleh AI.

CHAPTER 1: INTRODUCTION

1.1 Introduction

Waste is defined as any items that is discarded after primary use, considered worthless, defective and useless. It can be classified as compostable, general waste, and hazardous or non-hazardous waste based on characteristics like its toxicity, corrosiveness, explosiveness, and radioactivity (Wan Md Syukri Wan Mohamad Ali et al., 2015). Additionally, some waste categories are not widely recognized or understood by the general public, resulting in inefficient waste management.

Malaysia produces over 39,000 tons of waste every day, averaging around 1.17kg per person. The increase in production of waste is caused by factors like population growth, lifestyle changes, and the rapid urbanization. By 2050, Malaysia's landfills are projected to reach full capacity for waste disposal, raising concerns to the government (International Trade Administration, 2024). Waste can become a threat to the environment without a proper waste management. A strategic waste management is the solutions as the country has a lower recycling rate. Malaysia also experiences waste management challenges due to a lack of technology, experienced labour, and infrastructure to address the problem adequately (Saptaputra, 2023).Waste sorting helps to improve the recycling of these waste and reduces the amount of waste that ends up in the landfills.

Image recognition is a technology that uses machine learning to identify specific object in images. The application of image recognition systems to identify and classify waste has been explored in the article "Application of deep learning object classified to improve e-waste collection planning". This study do a research on how machine learning can make waste sorting easier by analyzing images submitted by users via smartphones (Saptaputra, 2023).

Therefore, this project introduces waste sorting app that uses image recognition to expand the waste sorting functions by simply uploading images to the application and be identified. All the submitted data will be tracked to monitor and analyze the users' habit over time.

1.2 Problem Statement/Research Problem

People have limited knowledge and understanding of waste categories. Many still have difficulty identifying them, even the common ones. Some are unaware that certain items, such as batteries, paints, and electronic components, must be disposed in a proper way as hazardous waste. The environmental impact of improper waste disposal and recycling contamination is often not well understood.

Despite the availability of waste sorting apps, effectively sorting compostable, hazardous, and general waste remains a challenge for many. Most of the current apps focus on identifying common recyclables, missing the opportunity to provide comprehensive guidance for all types of waste. These existing apps rarely come equipped with advanced features such as image recognition that would make the sorting process more intuitive and accessible. Some waste sorting apps use barcode scanner readers to identify items but it has limitations. This feature relies on the item information rather than the material type. Hence, items without barcode will not be recognized.

Recently, various automation strategies for waste sorting have been introduced, falling into three broad categories which are artificial intelligence (AI) approaches, Internet of Things (IoT) systems, and mechanical techniques. Mechanical methods rely on mechanical transfer, external detectors, and computer processors within an autonomous system to perform tasks similar to human waste-sorters. However, this approach faces challenges in waste

categorization due to inaccurate recognition and false detections. The integration of IoT devices with a cloud server has led to more accurate and automated waste sorting system, but it comes with high costs and complexity in setup and maintenance (Karrar Hameed Abdulkareem et al., 2024). On the other hand, Artificial Intelligence (AI) has the ability to learn and process complex data, whereas mechanical method and IoT systems rely solely on sensors or basic sorting mechanisms that cannot handle complex input. A more comprehensive app could fill these gaps by using AI-powered image recognition to identify types of waste, improving user engagement and promoting better disposal habits.

1.3 Objectives

The main objective of this project is to develop an application with the following objectives:

- To develop a waste sorting mobile application.
- To allow users to upload photo of waste items for automatic identification and categorization.

1.4 Scope

The scope of this project covers the following:

1. Development of a mobile application that can identify the following types of waste from images:
    - Compostable: Fruit peels, leaves, cardboard.
    - General: Plastic bottle, plastic bags, disposable cups.
    - Hazardous: Batteries, light bulbs, paint cans, mobile phones.
2. Allow users to capture and upload photos to identify the type of waste.
3. Implementation of features to track recycling activities and their outcomes.

1.5 Brief Methodology

Agile methodology is used in the making of this project. Agile focus on breaking down the development process into small, and manageable phases called sprints, allowing for frequent updates, adjustments, and continuous feedback. Agile welcomes change unlike other traditional models where plans are fixed and change is not welcomed. In Agile, activities like planning, designing, building, and testing are continuous and flexible rather than fixed (Challa Kaushik Koundinya et al., 2024). Figure 1 shows the Agile Planning Lifecycle.

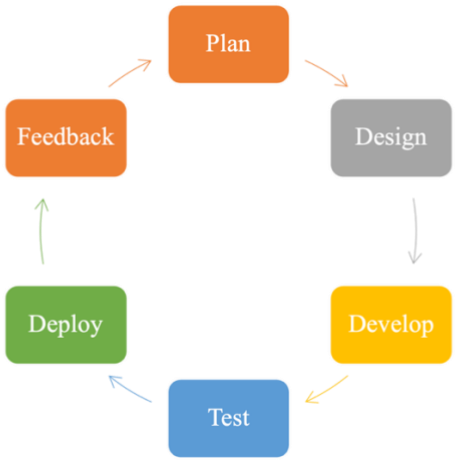


Figure 1.1: Agile Planning Lifecycle

Plan

Planning phase is where it starts with the breaking down of tasks into manageable parts, called sprints within a certain period. Each sprint focuses on specific tasks, like developing the image recognition feature and building the user interface within the planned time frame.

Design

During the design phase, the basic layout and structure of the app's functionality is created based on the previously feedback. The design evolves as the project moves forward, ensuring that each feature is aligned with the project's goals. This allows for frequent feedback and adjustments to the design. Decisions are made about which image recognition tools to use, the programming languages to use, data flow, algorithm design, and the user interface design.

Develop

The develop phase is where coding happens. Each sprint may involve creating parts of the app, like adding image recognition capabilities or setting up the user interface based on the design specifications. Each feature will be build step-by-step, focusing on core features and testing functionality as they go.

Test

After each feature is developed, tests are performed to ensure each feature works correctly as expected. For example, do a testing on the image recognition whether it accurately identifies the types of waste, and if the user interface is user-friendly and responsive. Testing is important in every sprint to ensure early detection and resolution of issues, creating an optimized version of the app.

Deploy

At the end of each sprint, a version of the app with newly added and improved features is place into operation. This version lets users see progress and give feedback, allowing for early adjustments and resolution of issues in the process.

Feedback

Feedback is gathered from users to determine any issues and features that need improvement. This ensures the requirements is met and the user experience is improved based on real-world usage.

The re-planning phase for the next iteration is based on the gathered feedback. The feedback is reviewed to determine which improvements should be included in the next iteration. Hence, the next sprint goal is adjusted and the project timeline is updated by adding new changes. Re-planning is ideal to keep the project flexible and adaptable by addressing changing requirements and continuously improving the product in each iteration.

1.6 Significance Of Project

Upon the completion of the Waste Sorting App with Image Recognition, users will be able to sort waste items by simply capturing and uploading photos to the application. Image recognition allows users to quickly sort waste, saving time in sorting and minimizing human error. The app acts as an educational tool, and with continued use it will help to raise awareness of proper waste disposable practices, reducing the risks of soil and water contamination by hazardous materials from general waste.

1.7 Project Schedule

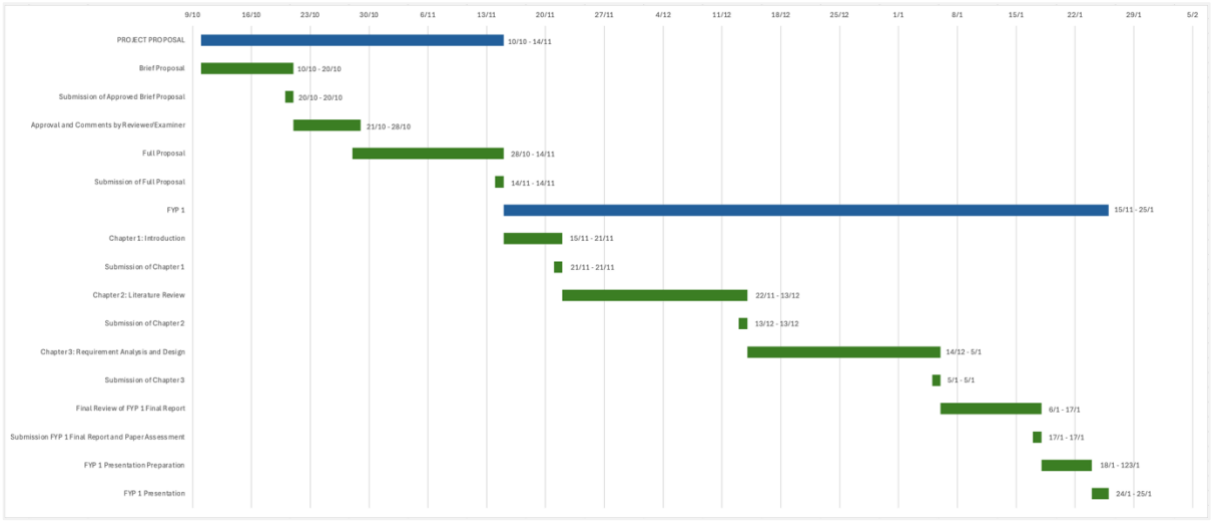


Figure 1.2: Gantt Chart 1

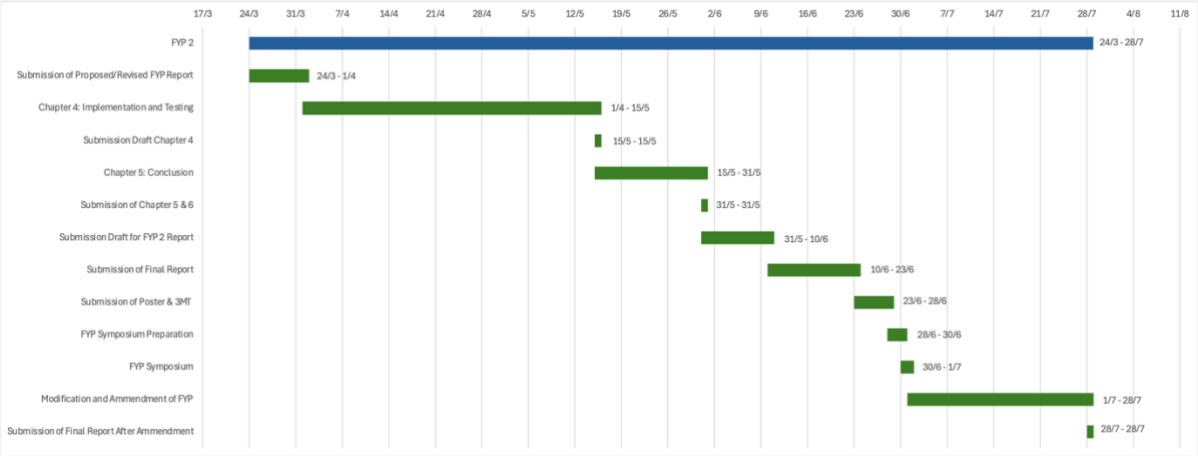


Figure 1.3: Gantt Chart 2

1.8 Expected Outcome

The project is expected to achieve the following outcomes by its completion:

i. A fully functional mobile application that accurately identify waste type using image recognition technology.

ii. Comprehensive analytics for users to track their waste disposal habits and measure progress over time.

iii. Features designed to track and record recycling activities to encourage continued eco-friendly practices.

iv. Source code repository with complete version for maintenance.

1.9 Thesis Outline

The report of this project is organized into five chapters as described below:

a) Chapter 1: Introduction

Chapter 1 begins with introduction, problem statements, project scope, and objectives. Then, a methodology is selected with the details of each step. Following by the significance of project, project schedule, and expected outcome of this project. All of the above provides an overview and introduction to the proposed application.

b) Chapter 2: Literature Review

Chapter 2 covers the background and a literature review of existing waste sorting applications. All relevant features and functions are examined and analyzed by comparing them to identify the most suitable features for the development of the proposed application.

c) Chapter 3: Requirement Analysis and Design

Chapter 3 covers the system architecture and the methodology used. Agile methodology is chosen which consists of six phases: plan, design, develop, test, deploy, and feedback. This chapter will discuss each of these phases in detail.

d) Chapter 4: Implementation and Testing

Chapter 4 outlines the implementation and testing phases of the project, developed according to the analyzed requirements. This chapter also includes screenshots of the developed application.

e) Chapter 5: Conclusion

Chapter 5 concludes the project by highlighting its achievements. It highlights potential future developments through recommendations and suggestions for further improvement.

1.10 Chapter Summary

This chapter introduces waste management issues in Malaysia, highlighting on the problems caused by improper waste disposal and the lack of understanding about different types of waste. While existing apps focus on recyclables and use barcode scanners, AI-powered image recognition that can identify a broader range of waste types provides a better solution

for waste sorting. The chapter explains the need for efficient waste sorting solutions and presents a mobile application designed to help users identify various types of waste. The application aims to make sorting waste easier, including able to track their recycling activities. This chapter also covers the scope, methodology (Agile), and expected outcomes.

CHAPTER 2: LITERATURE REVIEW

2.1 Introduction

This chapter discusses five similar existing applications that are closely related to the proposed waste sorting application. Each existing application will be analyzed and compared with the proposed application to highlight the similarities, differences, and opportunities for improvement. The comparison will focus on features, functions, and overall effectiveness to determine the best approach for achieving the objectives of the proposed application. The five similar existing applications are as follows:

a) Bower: Recycle & get rewarded
b) Recycling Assistant
c) EcoScan
d) BinSpect
e) WasteWizard

2.2 Review on Similar Existing Applications

2.2.1 Bower: Recycle & get rewarded

Bower is a cross-platform application available on both the Play Store and App Store, designed to encourage sustainable living through innovative waste management solutions. Users can earn rewards like coins by sorting and recycling waste. These rewards are backed by sponsorships from brands like Scan or Felix to promote increased recycling of packaging materials in Sweden and globally. Bower is an award-winner app, named one of Europe's top sustainability apps by Apple, and has received renowned achievements, including the Edie Awards 2024 and the Global Startup Awards 2023.

Bower's main features include identifying waste item through barcode scanning or image recognition. It also helps users locate nearby recycling or waste bins, making waste disposal convenient and accessible. Additional features include the ability to calculate the carbon dioxide emissions associated with their recycling efforts. Other than that, it can save recycling history and analyze detailed statistics, helping users track their progress over time.

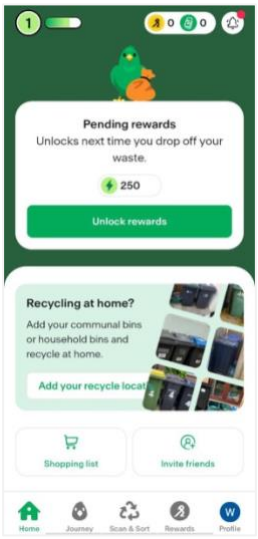


Figure 2.1: Home Screen of Bower

Figure 2.1 shows the home screen of the Bower app. At the top, a level bar is displayed alongside three icons representing reward, money, and notification. Below that, it shows the amount of pending rewards that will be unlocked after the next waste drop-off. Next, there is a section where users can add their recycle location. At the bottom, two buttons are displayed, one for accessing the shopping list and another for inviting friends.

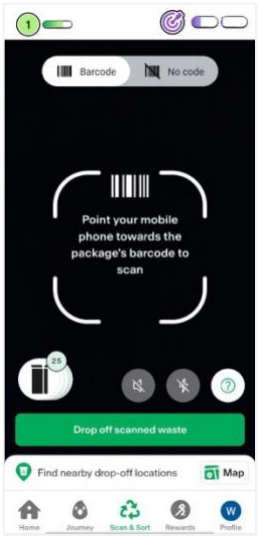


Figure 2.2: Barcode Scanner Feature in Bower

Figure 2.3: Image Scanner Feature in Bower

Figure 2.2 shows the barcode scanner feature, where users are required to point their phones' camera at the barcode on the waste. Figure 2.3 shows the image scanner feature, allowing users to capture an image of the waste directly using the camera.

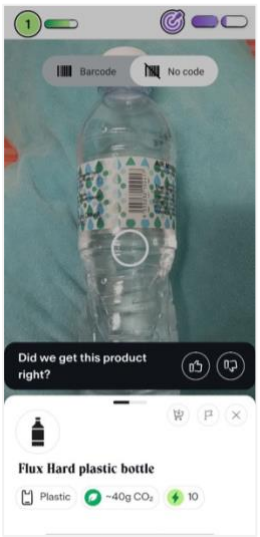


Figure 2.4: Scanning Results in Bower

Figure 2.4 displays the scanning results, showing the waste material and its type, along with the amount of carbon dioxide emissions and the number of XP points.

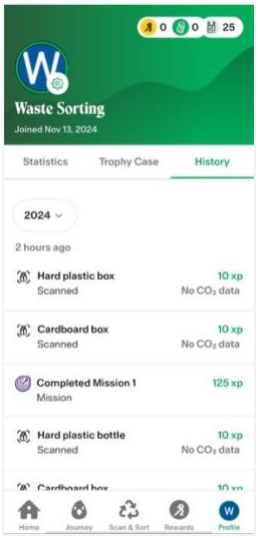


Figure 2.5: Profile Screen in Bower

Figure 2.5 shows the profile screen for Bower user. At the top, three icons are displayed, representing the number of rewards, money, and scanned items. Below these, the profile avatar, username and date joined are shown. The screen is divided into three tabs, which are statistics, trophy case, and history. The statistics tab displays recently dropped off waste, while the trophy case tab is for showcasing users' achievements. The history tab provides a complete record of recycling actions with details like waste type, XP earned, and CO2 emission data.

2.2.2 Recycling Assistant

Recycling Assistant is a cross-platform application available on both the Play Store and App Store, guiding users regarding certain types of waste. The types of waste like dry recyclables, bulky waste, food waste, general waste, glass, medication, plastic, special and hazardous waste, and textiles and clothes, can be detected through image recognition. English or a Ukrainian interface language are available to make recycling easier in Germany, Poland, and the Czech Republic.

Users can capture an image of waste with their phone's camera and the artificial intelligence-driven assistant will immediately determine the type of waste. The application is limited for Germany, Poland, United Kingdom, and the Czech Republic as the application will ask for users' location before using the app. All information, including disposal methods, and example of waste objects, is available within the application.

Figure 2.6: Home Screen of Recycling Assistant

Figure 2.6 shows the home screen of Recycling Assistant which is basically the sorting guide. It offers two tabs, which are categories and objects. The categories tab lists nine types of waste like dry recyclables, bulky waste, food waste, general waste, glass, medication, plastic, special and hazardous waste, and textiles and clothes. Each category leads to detailed information of the waste type as shown in Figure 2.7. The objects tab displays a list of waste objects as shown in Figure 2.8. At the bottom of the screen, a camera icon that allows users to access camera for scanning waste items.

Figure 2.7: Waste Information Screen in Recycling Assistant

Figure 2.7 shows the detailed information about a specific type of waste, including whether it is recyclable, a description of the waste type, examples of objects that fall under that type of waste, and ways on how to recycle them.

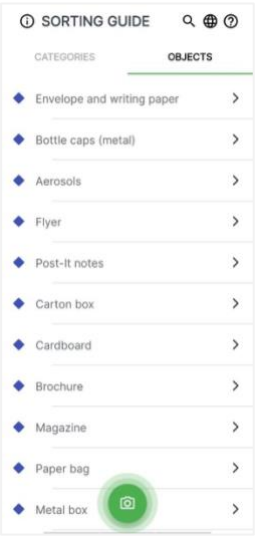


Figure 2.8: List of Waste Objects in Recycling Assistant

Figure 2.8 displays a list of waste objects, each of which leads to the detailed information about the waste and its type, as shown in Figure 2.7.

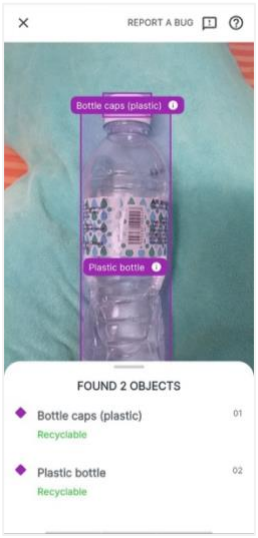


Figure 2.9: Scanning Results in Recycling Assistant

Figure 2.9 displays the scanning results, showing the waste material and its type, after capturing an image directly from phone's camera.

2.2.3 EcoScan

EcoScan is a Netherlands-based cross-platform application available on both the Play Store and App Store, covering a broad range of waste types, including paper, plastic, glass, metal, and electronics. The application gives clear and precise sorting advice, delivering personalized sorting advice based on location and local sorting rules. With a database of over 350,000 pieces of advice and an impressive 98% accuracy rate, EcoScan ensures users receive reliable information. Leveraging advanced scanning technology, it provides accurate and personalised recycling advice based user's location and recycling capabilities.

Users can take a photo using phone's camera or scan a barcode with the EcoScan app to receive guidance on the correct bin for their waste. *Litter Bingo* is a fun and interactive litter clean-up campaign in the participant's area, to make a positive environmental impact. By recycling waste, users are rewarded with EcoCoins, making

sustainability both engaging and rewarding. These EcoCoins can be used for discounts while buying eco-friendly products in the app's store.

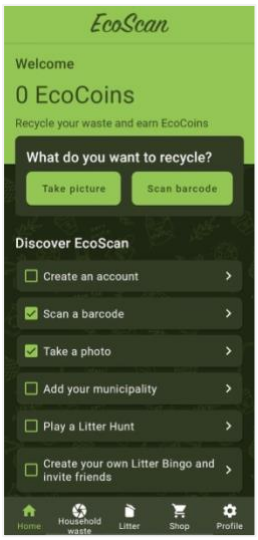


Figure 2.10: Home Screen of EcoScan

Figure 2.10 shows the home screen of EcoScan. At the top, the amount of EcoCoins, which can be earned by recycling, is displayed. Below that, two buttons provide options for scanning waste using either a barcode scanner or image scanner. Then, a list of features available is displayed, including creating an account, scanning barcodes, taking a photo, adding a municipality, playing Litter Hunt, and creating a Litter Bingo.

Figure 2.11: Item Analysis Result in EcoScan

Figure 2.11 shows the item analysis result, where users are required to select the correct analysis that were listed, which will lead to detailed information about the chosen waste type, as shown in Figure 2.12.

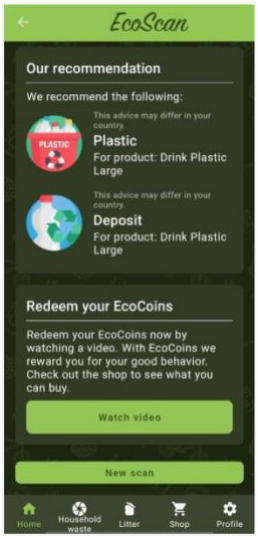


Figure 2.12: Waste Information Screen in EcoScan

Figure 2.12 presents the information about a specific type of waste, along with advices on how to handle it properly.

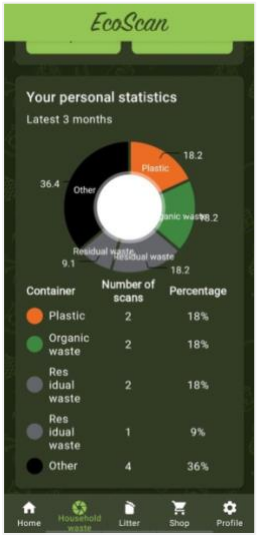


Figure 2.13: Personal Recycling Statistics in EcoScan

Figure 2.13 shows the personal recycling statistics for the past three months, presented through both pie chart and table. The pie chart visually represents the distribution of different waste types, with labels indicating the percentage of each category.

The table represents the number of scans for each waste type and their corresponding percentages.

2.2.4 BinSpect

BinSpect is an iOS application specifically designed to assist users in sorting out garbage without other big features. The application utilizes specialised image classifier machine learning model, classifying waste as either organic or inorganic. Hence, it has a little bit of limitations where the app unable to classify other type of waste.

Waste items can be scanned by capturing an image using phone's camera or selecting from the phone's gallery. After processing the provided image, it displays classification result along with the model's confidence level in percentage. Additionally, there is an option to save all the garbage's classification results in history.

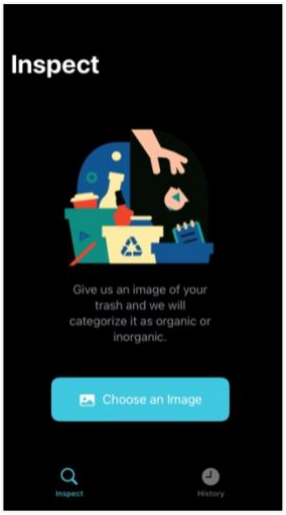


Figure 2.14: Main Screen of BinSpect

Figure 2.14 shows the main screen of BinSpect, displaying a large button that allows users to choose between capturing an image directly from camera or selecting an image from photo library.

Figure 2.15: Scanning Results in BinSpect

Figure 2.15 displays the scanning results, indicating whether the item is organic or inorganic, along with its confidence level. At the bottom, there are two buttons, one for saving the scanned data to the history and another for rescanning waste items.

Figure 2.16: History Screen in BinSpect

Figure 2.16 shows the history screen, which contains a list of scanned items. Each item card displays the waste type (organic or inorganic), the confidence level, and the date it was saved.

2.2.5 WasteWizard

WasteWizard is an iOS application developed that aims to reduce environmental pollution and waste mismanagement. The application classifies various types of trash like plastic, electronic waste (e-waste), batteries, metals, and biological waste. It leverages computer vision und machine learning to inform users how to dispose of a specific piece of trash. The waste item is identified and classified by capturing an image from phone's camera or choosing from the photo library.

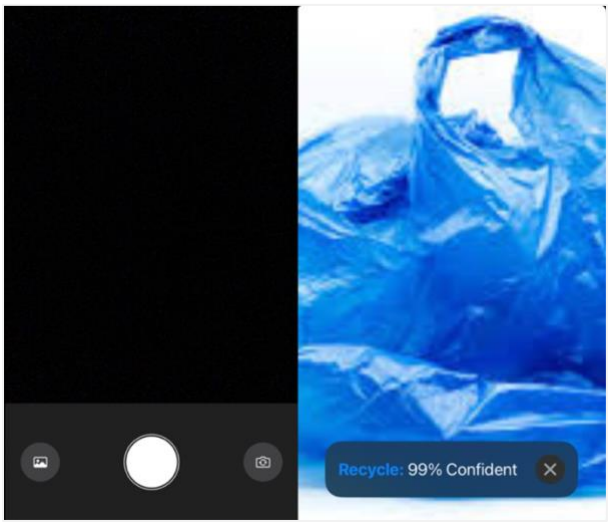


Figure 2.17: User Interface of WasteWizard

Figure 2.17 showcases the user interface of WasteWizard, which consists of the camera screen and the results after scanning. At the bottom, the results displays details like whether the item is recyclable, along with its confidence level.

2.3 Comparison Between the Existing Applications and Proposed Application

Table 2.1 shows the comparison of features between six existing applications: Bower, Recycling Assistant, EcoScan, BinSpect, WasteWizard, and the proposed Waste Sorting Application with Image Recognition. The table highlights the waste identification methods, where most applications use barcode scanners, image recognition, or combination of both,

while the proposed application focuses solely on image recognition. For image handling capability, some applications allow users to capture an image directly using camera, upload an image from gallery, or both. The proposed application supports both options. Regarding the tracking feature, which includes functionality like viewing history or statistics of sorted waste, only a few apps provide such capabilities, while the proposed application plans to enhance this by implementing interactive features that can track recycling activities and offer a more engaging experience.

Table 2.1: Comparison of Features Between Applications

Application	Waste Identification Method	Image Handling Capability	Tracking Feature	Additional Features
Bower: Recycle & get rewarded	Barcode scanner and image recognition	Camera Capture	History, statistics	Reward system
Recycling Assistant	Image recognition	Camera Capture	-	-
EcoScan	Barcode scanner and image recognition	Camera Capture	Statistics	-
BinSpect	Image recognition	Camera Capture /Upload from Gallery	History	-
WasteWizard	Image recognition	Camera Capture /Upload from Gallery	-	-
Proposed Application	Image recognition	Camera Capture /Upload from Gallery	History, statistics	Educational game, recycling calculator

As highlighted in Table 2.1, existing applications primarily rely on barcode scanning, and image recognition for waste identification. However, barcode scanning has notable limitations. This method relies on the item's information rather than the material type, resulting

in the inability to recognize items without barcodes. According to Amal Abdel-raouf et al. (2024), the article "Barcode-less Fruits Classification Using Deep Learning" discusses how deep learning techniques can classify objects without relying on barcodes, providing enhanced accuracy and efficiency. Advanced image recognition technology allows a more efficient waste identification process rather than barcode scanning.

Most of current existing applications are typically restricted to capturing images directly from the camera, which can be inconvenient for users who need to identify waste items that are not immediately accessible. This limitation can hinder the user experience, especially in scenarios when photos are sourced from other individuals or external parties like through social media. This limitation can be addressed by offering the flexibility to upload images from the gallery.

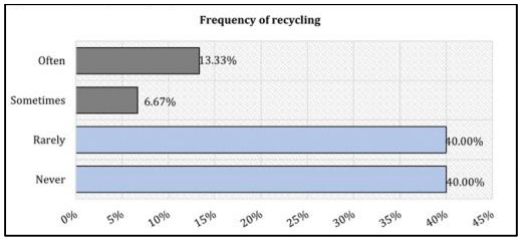


Figure 2.18: Frequency of Recycling

Figure 2.18 illustrates the participants' engagement in recycling behaviour before a recycling awareness program. The data reveals that 13.33% of participants reported recycling often, 6.67% sometimes, 40% rarely, and another 40% stated they never involved in recycling (Hafizah Hammad Ahmad Khan et al., 2024). These findings highlight the urgent need for solutions to motivate recycling behaviour, as a large majority of participants, approximately 80%, still have minimal level or zero level of engagement in recycling efforts.

Compared to existing applications, the proposed application offers a more comprehensive and engaging experience. While most apps support image recognition and basic tracking, this application goes a step further by implementing interactive features such as a

waste education game and a recycling value calculator, which are not commonly found in other solutions. In addition, the app includes a feedback and reporting feature that allows users to report inaccuracies or issues, which supports continuous improvement. By combining robust image recognition, multi-source image input, and user-centric tools, the app not only helps identify waste more accurately but also encourages better waste literacy and user engagement.

While existing applications provide tracking features like history and statistics, they often lack the depth required for users to effectively monitor their recycling efforts. Features that can enhance user motivation and give practical knowledge into the impact of their actions can play a crucial role in addressing the recycling issue. It makes recycling more accessible, and engaging, ultimately encouraging more people to adopt sustainable practices.

In a nutshell, the proposed application utilizes image recognition for efficient waste identification by allowing users to capture an image directly from the camera or upload from their photo gallery. With a comprehensive waste classification system that covers a broad range of waste types, the application includes features that enhance user motivation and offer practical knowledge into the impact of their actions, increasing engagement in recycling efforts.

2.4 Chapter Summary

This chapter analyzes all information regarding the waste sorting application by identifying the pros and cons of each application. After reviewing and comparing five (5) similar existing applications with the proposed application, a decision on the inclusion of features has been made. The analysis highlights key differentiators like waste identification methods, image handling capabilities, tracking features, platform compatibility, and waste classification categories. By identifying gaps in current solutions, this chapter provides a

foundation for a more effective and user-friendly waste sorting application with image recognition that addresses existing challenges in waste management.

CHAPTER 3: REQUIREMENT ANALYSIS AND DESIGN

3.1 Introduction

The background study and evaluation of existing waste sorting applications have led to a clear definition of the requirements which will be discussed in this chapter. It begins with a detailed breakdown of the requirements that ensures the app's design aligns with both user needs and technical feasibility. The iterative design and planning process are explored, highlighting the system architecture and logical design through various diagrams. Finally, the physical design is presented to illustrate the app's user interfaces. This structured approach ensures a seamless transition from planning to action.

3.2 Agile Methodology

Agile methodology is used in the development of the Waste Sorting App, encouraging adaptability to changes in user requirements and emerging technologies. Figure 3.1 illustrates the Agile Planning Lifecycle.

Figure 3.1: Agile Planning Lifecycle

During planning phase, it begins by breaking down tasks into manageable sprints, each focusing on specific objectives relevant to the Waste Sorting App. For example, one sprint may concentrate on developing the image recognition feature, while another focuses on enhancing the user interface.

During the design phase, the layout and functionality are developed based on insights from existing waste sorting applications. This iterative process ensures that features align with project objectives and user requirements. Key decisions involve selecting image recognition tools, programming languages, and creating a user-friendly interface.

The development phase involves the actual coding of the app. Each sprint includes tasks like adding image recognition features or building the user interface based on the design. Features are created step-by-step and ongoing testing is done during this phase to make sure each part works as expected, helping to find and fix problems early on.

After the completion of each feature, thorough testing is conducted to check that everything works correctly. For instance, the app's image recognition feature is tested to ensure it accurately identifies various types of waste.

At the end of each sprint, a new version of the app, with added and improved features, is deployed for user evaluation. This gives users a chance to see the app's progress and provide feedback to make early changes and fix any problems that come up.

User feedback is systematically gathered to identify any problems and areas for improvement. This ensures that the app meets user requirements and enhances the overall user experience based on real-world usage. The re-planning phase for the next iteration relies on this collected feedback to prioritize which improvements should be implemented in the upcoming sprint. Consequently, the goals for the next sprint are adjusted, and the project timeline is updated to reflect new changes.

3.3 Requirement Analysis

The requirement analysis focuses on defining the key requirements and specifications for the development of the proposed application.

3.3.1 User Requirements

The proposed application serves for 2 main user groups, general users and administrators. General users are the one who interact with the app, while administrators are system users with the authority to manage the data and user management. The user requirements are gathered from the analysis of survey conducted and the results are as in the figures below.

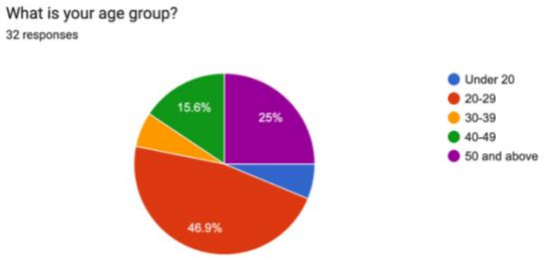


Figure 3.2: Data Result on Respondents' Age Group

Figure 3.2 shows a pie chart that reveals people of all age groups can utilize the proposed application. Taking consideration that the users are from a wide range of age group, the user interface must be user-friendly.

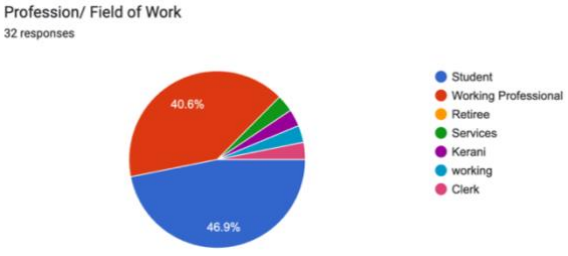


Figure 3.3: Data Result on Respondents' Employment Status

Figure 3.3 shows a pie chart illustrating the distribution of respondents across different professions. This diversity suggests that the app's potential user base spans a wide range of occupations.

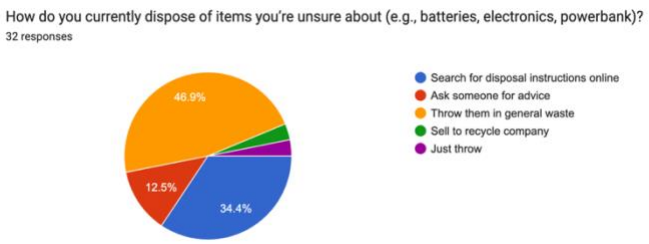


Figure 3.4: Data Result on Respondents' Current Disposal Practices for Uncertain Items

Figure 3.4 shows that 46.9% of respondents throw items they are unsure about, like batteries, electronics, and powerbanks into general waste. This is a concerning as these items contain hazardous materials that can leach into the environment and pose risks to human health and ecosystems.

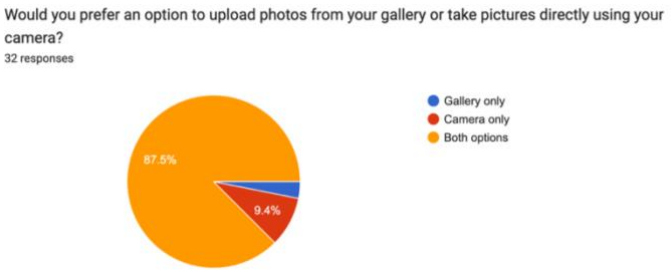


Figure 3.5: Data Result on Respondents' Preference of Uploading Image

Figure 3.5 shows that a significant majority of respondents would like to have both options of uploading photos from their gallery and taking pictures directly from camera. This indicates a strong preference for flexibility and convenience in photo management.

Figure 3.6: Data Result on Respondents' Expectations Information in Scanning Results

Figure 3.6 shows that a high percentage of respondents expect to see both the "Waste type" and "Disposal instructions" in the scanning results. This indicates a strong desire for clear and actionable information to guide their waste sorting decisions.

Figure 3.7: Data Result on Respondents' Response of History Feature

Figure 3.7 shows that a significant majority find the idea of tracking their waste sorting history in the app either "somewhat useful" or "very useful". This indicates a strong interest in using the app to monitor their progress and track their contribution to waste reduction efforts.

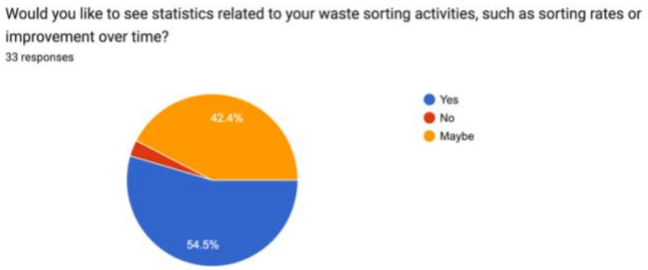


Figure 3.8: Data Result on Respondents' Response of Tracking Statistics Feature

Figure 3.8 shows that a majority of respondents express interest in viewing statistics which indicates a desire for data-driven insights into their waste management habits and progress that can help motivate users in their waste reduction efforts.

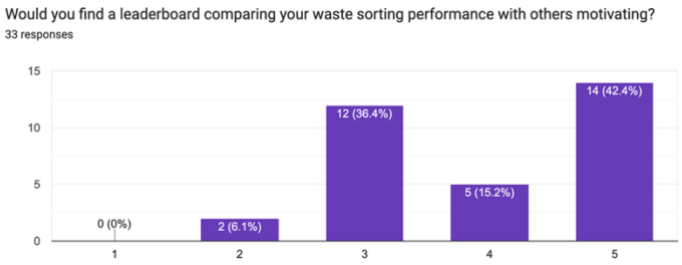


Figure 3.9: Data Result on Respondents' Response to Leaderboard Feature

Figure 3.9 shows that a significant portion of respondents find the idea of a leaderboard comparing their waste sorting performance with others very motivating. This suggests that gamification and social comparison could serve as a fun and engaging way to encourage users to improve their waste sorting habits.

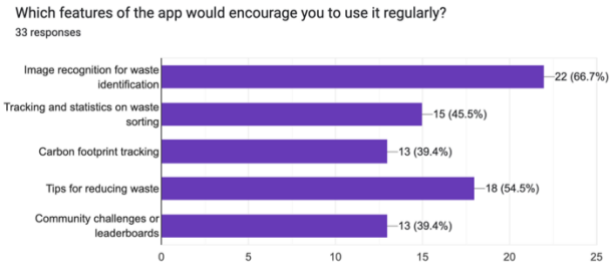


Figure 3.10: Data Result on Respondents' Preferences for App Feature

Figure 3.10 shows that the most encouraging feature for users to use the app regularly is image recognition for waste identification**,** which is the main features of the app. This suggests that the app's core functionality of accurately identifying waste types is a key driver of user engagement.

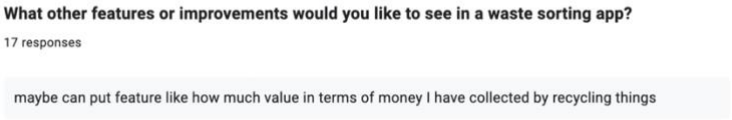


Figure 3.11: Data Result on Respondents' Suggestions

Figure 3.11 shows a suggestion from a respondent that seems can be included in the app. This feature can give motivation to user to use the app as it relates with money. From the results of the survey, these requirements are aligned with the project's objectives.

The app needs to have a simple and intuitive design for an easy navigation for all age groups to reach a wider audience. The features that included are both options of uploading images by capturing image directly from their device camera or selecting images from their photo gallery for identification of waste item, include waste type and disposal instructions in the scanning results, history, statistics tracker, and some activities, and performance on a leaderboard compared to other users.

3.3.2 Software Requirements

Software requirements refer to the necessary frameworks, software tools, and applications needed to develop, operate, and use the proposed application effectively. These requirements ensure the app's functionality, user experience, and overall system performance.

Table 3.1: Software Requirements for Waste Sorting App with Image Recognition

Environment	Requirement	Description
Development	Framework	React Native
	APIs	Firebase, Google Cloud Vision API
	Database	Firebase Firestore
	Development Tools	Visual Studio Code, Android Studio
	Version Control	Git, GitHub
Delivery	Operating System	• Android **8.0 (Oreo)** and above • iOS **12.0** and above
	Applications	• Camera app (for scanning and identifying waste) • Email app (for account creation and verification)
	Web Browser	Google Chrome, Safari, Firefox, Microsoft Edge, etc. (for accessing app additional resources)

Table 3.1 outlines the essential software requirements for the proposed application. They are categorized into development and delivery phases. Development phase is the requirements needed to build and maintain the app, while delivery phase includes software needed to access the app with its features. In the development phase, React Native is used as the framework for frontend development, Firebase as the backend, and Google Cloud Vision API for the image recognition functionalities which leveraging its advanced machine learning algorithms for accurate object detection and labelling. Firestore serves as the database, tools like Visual Studio Code and Android Studio are used in coding and testing, and Git and GitHub for collaboration. For the delivery phase, users need to access the app with Android or iOS. Users are required to have a camera app, a social media app optionally, and email app for using the app's features. A web browser is also needed to access app additional resources that might be added into the app.

3.3.3 Hardware Requirements

Hardware requirements include the physical devices and specifications needed to run the app. Table 3.2 outlines the hardware requirements which consist of three requirements.

Table 3.2: Hardware Requirements for Waste Sorting App with Image Recognition

Requirement	Description
Device Compatibility	Smartphone or tablet with at least 2GB RAM and a 2GHz processor
Camera	Minimum resolution of 8 MP (for capturing high-resolution images to ensure that the Google Vision API can accurately analyze waste items)
Internet Connectivity	Stable internet connection

Processing Power	Devices with a sufficient processing power (for handling image capture and display results smoothly while the heavy lifting will be done in the cloud)

3.3.4 Functional Requirements

Functional requirements outline the core functionalities and operations of the proposed application to ensure that the app delivers its objectives effectively based on the user requirements stated above. The function requirements consist of:

a) Object Recognition

- The Google Vision API must analyze the uploaded images to return recognized waste categories with confidence scores.

b) Image Uploading

- Upload images to Firebase Storage for processing by the Google Vision API.

c) Sorting Logic Module

- Processes the recognized waste data and determines sorting criteria based on predefined rules.

d) Waste Activities

- Provide features such as a gamification, leaderboard, and recycling value calculator to motivate and engage users in waste management activities.

e) Search, Filter, and Sort

- Enable users to efficiently locate data by providing search, filter, and sorting options for easier access and organization.

f) User Accounts

- Enable users to create accounts to track their waste management progress.

g) Feedback System

- Allow users to submit issues or suggestions directly through the app.

3.3.5 AI Algorithms

Google Cloud Vision is one of the most used widely automated system with commercial cloud services for advanced image analysis using machine learning (Karanrat Thammarak et al., 2022). Vision API processes visual data to extract information from an image. It offers various features like image labelling, face and landmark detection, optical character recognition (OCR), object localization, and tagging of explicit content. Among these, image labelling and object localization are particularly relevant for the proposed waste sorting application.

a) Image Labelling

b) Object Localization

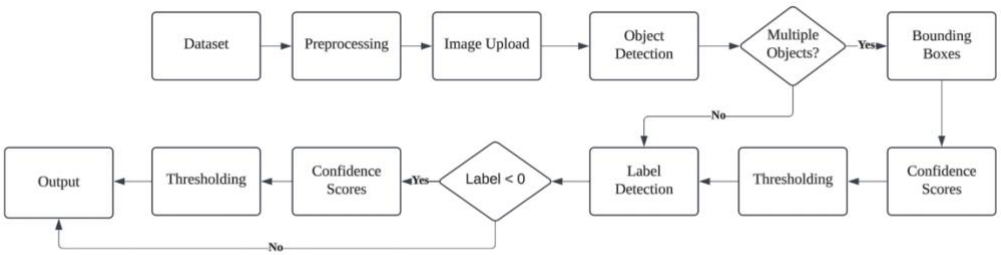


Figure 3.12: Flowchart for Image Analysis

Figure 3.12 shows the flow of image analysis process. It starts by taking a dataset of waste images and preparing them for analysis. First, the uploaded images go through object detection to find and locate objects in the images. If multiple objects are found, the system draws boxes around them and filters out low-quality detections based on confidence scores. If no objects are detected, this step is skipped, and the process moves directly to label detection. In label detection, the system extracts information about the

objects in the images. If labels are successfully detected, confidence scores are applied to filter the results and generates output based on the available information. If not, the process moves directly to the output stage. Thus, at the end, the output is generated, which may include detected objects with their labels.

3.3.6 Machine Learning Model

Google Gemini has the ability of managing multiple forms of data where it can process and generate content using text, images, audio, videos, and even PDFs (Muhammad Imran & Norah Almusharraf, 2024). In the context of the waste sorting application, this means the model can analyze images of waste, identify objects, and classify them into categories. Gemini models are trained on a dataset that includes data from web documents, books, and code, and various form of media (Anil, R. et al., 2023). Hence, the model can interpret waste labels and recognize text on packaging, and even differentiate between similar-looking waste materials. By implementing Gemini's advanced image labelling and object localization capabilities, it stands out as the ideal choice for the proposed application due to its adaptability to the real-world scenarios.

3.4 Iterative Design and Planning

This section outlines the ongoing process of designing and refining the app's structure and features through multiple iterations. It is to ensure continuous improvement based on feedback and requirements.

3.4.1 System Architecture

Figure 3.13 illustrates the architecture of the proposed application designed to describe the high-level structure of the app, including how different components interact and integrate.

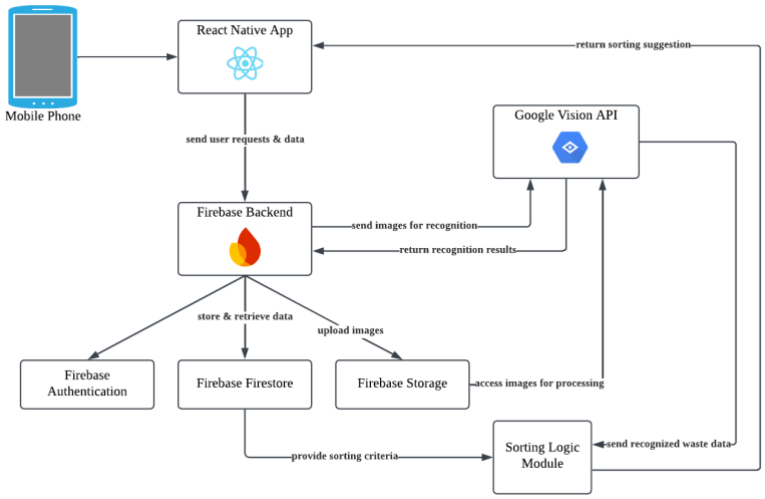


Figure 3.13: System Architecture of Waste Sorting App with Image Recognition

Firstly, ensure that the user has installed the app in their devices. Then, the user captures an image of the waste item directly from the camera or select from the gallery. This image is then uploaded to Firebase Storage, a cloud-based storage service. From there, the image is sent to Google Vision API for object recognition. The Google Vision API analyzes the image and identifies the objects, such as plastic bottles, cardboard, or batteries. The recognized waste data is then stored in Firebase Firestore, a NoSQL database. Next, the Sorting Logic Module processes the recognized waste data that uses

predefined sorting criteria to determine the waste types, and other relevant factors. Finally, the app displays the sorting results to the user.

3.4.2 Logical Design

Logical design focuses on the conceptual framework of the app, detailing data flow, relationships, and processes to ensure logical functionality.

3.4.2.1 Use Case Diagram

Figure 3.14 illustrates the interactions between users which are general user and administrator with the app by defining and organizing various functional requirements in the proposed application.

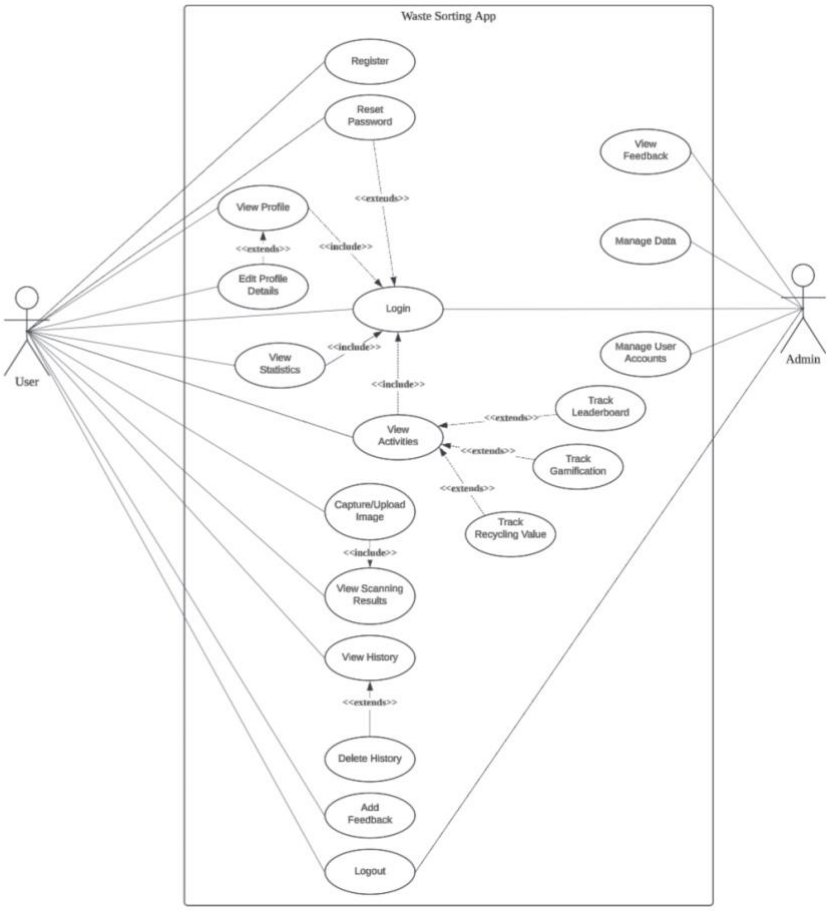


Figure 3.14: Use Case Diagram

Table 3.3 to Table 3.21 outlines descriptions of each use case to identify potential gaps in functionality, allowing adjustment before final deployment. Each table includes the use case name, short description, actors involved, preconditions, postconditions, steps involved in executing the use case and the alternative and exception flows detail any variations or errors that may occur during execution.

Table 3.3 shows the use case description for "Register" functionality on how it allows the user to create an account.

Table 3.3: Use Case Description for Register

Use Case	Register
Short Description	Allow the user to create an account.
Actor(s)	User
Precondition(s)	The user has installed the app but not registered.
Postcondition(s)	• Upon successful registration, a new user account is created and stored in the database. • Upon unsuccessful registration, no new user account is created.
Main Flow	1. The user navigates to the registration page. 2. The user provides required information like first name, last name, email, password. 3. The system validates all provided information like check for empty fields, email format, password complexity, password matching, check for existing account with the same email. 4. The user submits the form. 5. A newly created user account is added to the database. 6. A success notification is displayed. 7. The user is redirected to the login page.
Alternative Flow	(A1) User chooses google login:

	1. The user selects google login option by clicking on the “Login with Google” button. 2. The system redirects the user to the google login provider’s authentication page. 3. The user authenticates with the google login provider. 4. The system retrieves user information from the google login provider. 5. A newly created user account is added to the database. 6. A success notification is displayed. 7. The user is redirected into the app. (A4) Existing account found: 1. The system detects that the provided email is already associated with an existing account. 2. The system displays an error message informing the user that the email is already taken. 3. The user is prompted to choose a different email or reset password to recover access to the account. 4. The user fixes the issue and proceeds with registration. (A5) Missing or invalid input data: 1. The user provides invalid data or empty input data. 2. The system displays certain error messages for the invalid fields. 3. The user is prompted to correct the invalid data. 4. The user fixes the issue and proceeds with registration.
Exception Flow	(E1) Existing account found: 1. The system detects that the provided email is already associated with an existing account. 2. The system displays an error message informing the user that the email is already taken. 3. The user is prompted to choose a different email or reset password to recover access to the account.

	4. The user cannot fix the issue as the email is not active and unreachable by the user. (E3) User cancels registration: At any point during the registration process, the user may choose to cancel and exit the registration flow.

Table 3.4 shows the use case description for "Login" functionality on how it allows the user and admin to log into the app.

Table 3.4: Use Case Description for Login

Use Case	Login
Short Description	Allow the user/admin to access the app.
Actors(s)	User, Admin
Precondition(s)	• The user/admin must have an account. • The user/admin must have valid login credentials.
Postcondition(s)	• Upon successful login, the user/admin is redirected to the app interface and has access to the exclusive functionalities. • Upon unsuccessful login, the user/admin is not logged into the app and the access to the exclusive functionalities is denied.
Main Flow	1. The user/admin navigates to the login page. 2. The user/admin provides their email/username and password. 3. The system checks if the provided credentials match the stored user/admin data in the database. 4. If the credentials are valid, the user/admin will be logged into the app. 5. A success notification is displayed.
Alternate Flow	(A1) User chooses google login:

	1. The user selects google login option by clicking on the “Login with Google” button. 2. The system redirects the user to the google login provider’s authentication page. 3. The user authenticates with the google login provider. 4. The system retrieves user information from the google login provider. 5. The user is logged into the app. (A2) Missing or invalid input data: 1. The user/admin provides invalid data or empty input data. 2. The system displays certain error messages for the invalid fields. 3. The user/admin is prompted to correct the invalid data. 4. The user/admin fixes the issue and proceeds with login.
Exception Flow	(E1) Account locked: 1. The user has multiple failed login attempts. 2. The user’s account is locked. 3. The system displays an error message informing the user that their account is locked. 4. The user might need to contact the developers to fix the issues.

Table 3.5 shows the use case description for “Reset Password” functionality on how it allows the user to reset their password.

Table 3.5: Use Case Description for Reset Password

Use Case	Reset Password
Description	Allow the user to reset their password if forgotten.
Actors	User
Preconditions	• The user must have an account.

	• The user must have an access to their registered email inbox.
Postconditions	• Upon successful password reset, the password is successfully updated and can log in with new password. • Upon unsuccessful password reset, the password remain unchanged.
Main Flow	1. The user clicks “Forgot Password” on the login page. 2. The system prompts the user to enter their email associated with their account. 3. The system validates the provided email. 4. The system sends a reset link through email. 5. The user enters new password in the required field and submit. 6. The system updates the user’s password with new value in the database. 7. The user is redirected to the login page.
Alternate Flow	(A1) Reset link not received: 1. The user does not receive the reset link. 2. The user requests a new one by re-entering the form. 3. The system sends a new reset link. 4. The user enters the new password and submit. 5. The user proceeds with the reset password. (A2) Missing or invalid input data: 1. The user provides invalid data or empty input data. 2. The system displays certain error messages for the invalid fields. 3. The user is prompted to correct the invalid data. 4. The user fixes the issue and proceeds with reset password.
Exception Flow	(E1) Reset link not received: 1. The user does not receive the reset link. 2. The user requests a new one by re-entering the form.

	3. The system sends a new reset link. 4. The user enters the new password and submit. 5. The user proceeds with the reset password. (E2) No existing account found: 1. The system detects that the provided email is not associated with any existing account. 2. The system displays an error message informing the user that the email is not registered. 3. The user cannot fix the issue as the email does not exist in the database. (E3) User cancels reset password: At any point during the process, the user may choose to cancel and exit the reset password flow.

Table 3.6 shows the use case description for "View Profile" functionality on how it allows the user to view their profile.

Table 3.6: Use Case Description for View Profile

Use Case	View Profile
Description	Allow the user to view their profile details.
Actors	User
Preconditions	• The user must have an account. • The user must be logged in.
Postconditions	The user can view their profile details.
Main Flow	1. The user navigates to the profile page. 2. The system retrieves the user's profile details from the database. 3. The system displays the information.
Alternate Flow	N/A
Exception Flow	N/A

Table 3.7 shows the use case description for “Edit Profile Details” functionality on how it allows the user to edit their profile details.

Table 3.7: Use Case Description for Edit Profile Details

Use Case	Edit Profile Details
Description	Allow the user to edit their profile details.
Actors	User
Preconditions	• The user must have an account. • The user must be logged in. • The user must be able to view the all the information of their profile.
Postconditions	• Upon successful update, the user’s profile details are updated in the database. • Upon unsuccessful update, the user’s profile details remain unchanged in the database.
Main Flow	1. The user navigates to the edit profile page by clicking on the “Edit Profile” button in profile page. 2. The user edits the details (first name, last name, gender, email, phone number, date of birth, profile picture) by providing new value. 3. The system validates the new value input and saves the updated details in the database. 4. A success notification is displayed. 5. The user can view the updated profile details.
Alternate Flow	(A1) Missing or invalid input data: 1. The user provides invalid data or empty input data. 2. The system displays certain error messages for the invalid fields. 3. The user is prompted to correct the invalid data. 4. The user fixes the issue and proceeds with editing.
Exception Flow	(E1) User cancels edit: At any point during the editing process, the user may choose to cancel and exit the editing flow.

Table 3.8 shows the use case description for “Capture/Upload Image” functionality on how it allows the user to capture and upload image of waste.

Table 3.8: Use Case Description for Capture/Upload Image

Use Case	Capture/Upload Image
Description	Allow the user to capture and upload an image of waste to the scanner for sorting.
Actors	User
Preconditions	• The user has access to a device with a camera and sufficient storage. • The user has the necessary permissions to access the device’s camera and storage.
Postconditions	• Upon successful item scanning, the image of the waste item is captured or uploaded and proceed with the next sorting process. • Upon unsuccessful item scanning, an error message is displayed and the user is required to re-capture or re-upload an image.
Main Flow	1. The user navigates to the Scan page. 2. The user chooses between the options of capturing image direct from the camera or uploading an existing image from the device. 3. If capture option is selected, the system opens the device camera and the user captures the image of the waste item. If upload option is selected, the system opens the file picker and the user chooses the desired image from their device. The user can crop their selected image before uploading. 4. The system uploads and processes the image. 5. The system displays the results of the recognized waste type with its information.

	6. The user can choose to scan another item or exit the scanning process.
Alternate Flow	(A1) Image processing error: 1. The user confirms the image for scanning. 2. The system encounters an error while processing the image. 3. The system displays an error message indicating that the image could not be processed. 4. The user can try re-capturing the image or re-uploading a different image.
Exception Flow	(E1) User interruption: 1. The user interrupts the image capture/upload process by closing the app. 2. The process is cancelled. (E2) User cancels image capture/upload: At any point during the process, the user may choose to cancel and exit.

Table 3.9 shows the use case description for "View Scanning Results" functionality to display the result of the scanned waste.

Table 3.9: Use Case Description for View Scanning Results

Use Case	View Scanning Results
Description	Display the result of scanned waste.
Actors	User
Preconditions	A scan has been completed successfully.
Postconditions	The user can view the result of waste sorting with its details.
Main Flow	1. After the scanning process is completed, the system displays the scanning results of the uploaded image including recognized waste type and other additional information. 2. The user reviews the scanning results for accuracy and information.

	3. The user saves the results directly to history. 4. The user completes their review of the scanning results and takes desired actions.
Alternate Flow	(A1) Incorrect Results: 1. The user reviews the results and believes the recognized waste type is incorrect. 2. The user clicks the "Report Issue" icon. 3. The system navigates the user to the feedback page for providing feedback about the incorrect recognition. 4. The user submits their feedback. (A2) Share scanning results: 1. The user wants to share the scanning results. 2. The user clicks the share icon and share via social media.
Exception Flow	(E1) Results Not Available: 1. The user attempts to view the scanning results. 2. The system encounters an error retrieving the results. 3. The user can return to the home screen or contact the developers to fix the issues.

Table 3.10 shows the use case description for "View History" functionality on how it allows user to view their history.

Table 3.10: Use Case Description for View History

Use Case	View History
Description	Allow the user to view a history of their scanned waste.
Actors	User
Preconditions	• The user has scanned a waste item before. • The app has the necessary data stored.
Postconditions	The user can view the history of scanned and sorted waste item throughout the usage of the app.

Main Flow	1. The user clicks history button at the bottom bar and navigates to the history page. 2. The system retrieves a list of previously scanned waste items from the database and displays them to the user.
Alternate Flow	(A1) Filter history: 1. The user wants to filter the scan history by date range, waste type, or other criteria. 2. The user selects the filter options and applies them. 3. The system updates the history list based on the selected filter options. (A2) Sort history: 1. The user wants to sort the scan history by date, waste type, alphabetically, or other parameters. 2. The user selects the sort options and applies them. 3. The system updates the history list based on the selected sort options. (A3) Search history: 1. The user wants to search the scan history by waste type or waste name and provides them in the required field. 2. The system updates the history list based on the provided value.
Exception Flow	(E1) No History Available: 1. The user navigates to the history page. 2. No items are displayed indicating that no scans have been performed. 3. The user is prompted to scan waste items to build their history.

Table 3.11 shows the use case description for “Delete History” functionality on how it allows user to delete a specific or all history entries.

Table 3.11: Use Case Description for Delete History

Use Case	Delete History
Description	Allow the user to delete the history of their scanned waste.
Actors	User
Preconditions	• The user has scanned a waste item before. • The app has the necessary data stored.
Postconditions	The user can delete specific scanned waste item in the history.
Main Flow	1. The user navigates to the history page and views the list of previously scanned waste items. 2. The user selects the specific scanned waste item(s) they want to delete by clicking on the delete icon on bottom right of the item card. 3. The user confirms the deletion action. 4. The system deletes the selected scan history entries from the database. 5. A success notification is displayed.
Alternate Flow	N/A
Exception Flow	(E1) User cancels deletion: At any point during the deletion process, the user may choose to cancel and exit the deletion flow.

Table 3.12 shows the use case description for “View Statistics” functionality on how it allows user to view waste sorting statistics.

Table 3.12: Use Case Description for View Statistics

Use Case	View Statistics
Description	Allow the user to view waste sorting statistics.
Actors	User
Preconditions	• The user must have an account. • The user must be logged in. • The user has scanned a waste item before. • The app has the necessary data stored.

Postconditions	The user can view the generated statistics.
Main Flow	1. The user navigates to the statistics page. 2. The system retrieves statistical data related to the total waste items scanned and waste types recognized. 3. The user can view the graphs of the statistical data.
Alternate Flow	(A1) Filter statistics: 1. The user wants to filter the displayed statistics. 2. The user selects the filter options by date range (daily, weekly, monthly, yearly) and applies them. 3. The system updates the statistics based on the selected filter options. (A2) Share statistics: 1. The user wants to share the displayed statistics. 2. The user clicks the share icon and share via social media.
Exception Flow	(E1) No Statistics Available: 1. The user navigates to the statistics page. 2. No statistical data is displayed indicating no scans have been performed. 3. The user is prompted to scan waste items to generate statistics.

Table 3.13 shows the use case description for "View Activities" functionality on how it allows user to view the waste sorting activities.

Table 3.13: Use Case Description for View Activities

Use Case	View Activities
Description	Allow the user to view the waste sorting activities such as leaderboard, gamification, and recycling value calculator.
Actors	User
Preconditions	• The user must have an account. • The user must be logged in.

	• The user must collect points to unlock each activity, based on the following minimum requirements: o Leaderboard: 5 points or more o Gamification: 20 points or more o Recycling Value Calculator: 40 points or more
Postconditions	The user can view the list of activities with their details based on the user's membership tier: • Leaderboard: Available for users with 5 points or more (Bronze members). • Gamification: Available for users with 20 points or more (Silver members). • Recycling Value Calculator: Available for users with 40 points or more (Gold members).
Main Flow	1. The user navigates to the activity page by clicking on the activity icon in the bottom bar. 2. The system checks the user's account and retrieves their total points. 3. The system displays a list of activities based on the user's membership tier: a. Leaderboard (5+ points) b. Gamification (20+ points) c. Recycling Value Calculator (40 + points) 4. The user selects the activity they want to view. 5. The system displays all relevant details for the selected activity.
Alternate Flow	N/A
Exception Flow	(E1) Insufficient points: 1. The user attempts to select an activity for which they do not meet the required points. 2. The user unable to click on the selected activity as it has not been unlocked yet.

	3. The user is encouraged to scan waste and start earning points.

Table 3.14 shows the use case description for “Track Leaderboard” functionality on how to monitor user leaderboard based on activity performance.

Table 3.14: Use Case Description for Track Leaderboard

Use Case	Track Leaderboard
Description	Monitor user leaderboard based on activity performance.
Actors	User
Preconditions	• The user must have an account. • The user must be logged in. • The user must have at least 5 points (Bronze tier).
Postconditions	The user can view the leaderboard.
Main Flow	1. The user selects “Leaderboard” from the activity page. 2. The system fetches all users’ data. 3. The system processes the data and calculates the user rankings based on their total points. 4. The system displays the user’s rank relative to others.
Alternate Flow	N/A
Exception Flow	(E1) Insufficient points: 1. The user attempts to access the Leaderboard without reaching the required 5 points. 2. The user unable to click on the selected activity as it has not been unlocked yet. 3. The user is encouraged to scan waste and start earning points.

Table 3.15 shows the use case description for “Track Gamification” functionality on how to enjoy the gamification.

Table 3.15: Use Case Description for Track Gamification

Use Case	Track Gamification
Description	Enjoy a gamified quiz-based activity to test.
Actors	User
Preconditions	• The user must have an account. • The user must be logged in. • The user must have at least 20 points (Silver tier).
Postconditions	The user is able to access and play the gamified "Know Your Trash" activity.
Main Flow	1. The user selects “Know Your Trash” from the activity page. 2. The system verifies the user’s tier and point eligibility. 3. The system loads a set of randomized quiz questions related to waste sorting. 4. The user selects answers and progresses through the quiz. 5. The system evaluates the answers and provides feedback after each question. 6. Upon completion, the system displays the score.
Alternate Flow	N/A
Exception Flow	(E1) Insufficient points: 1. The user attempts to access the Gamification without reaching the required 20 points. 2. The user unable to click on the selected activity as it has not been unlocked yet. 3. The user is encouraged to scan waste and start earning points.

Table 3.16 shows the use case description for “Track Recycling Value” functionality on how to monitor total recycling value contributed by the user.

Table 3.16: Use Case Description for Track Recycling Value

Use Case	Track Recycling Value
Description	Monitor the total recycling value contributed by the user.
Actors	User
Preconditions	• The user must have an account. • The user must be logged in. • The user must have at least 40 points (Gold tier).
Postconditions	The user can view their overall recycling value.
Main Flow	1. The user selects “Recycling Value Calculator” from the activity page. 2. The system fetches the user’s waste-scanning history including types of waste scanned and quantities of each waste type. 3. The system fetches current market rates for each waste type like price per kilogram. 4. The system processes the data and calculates the total monetary value of the user’s waste collection. 5. The system displays the total monetary value of the waste they can sell and breakdown of value by waste type.
Alternate Flow	N/A
Exception Flow	(E1) Insufficient points: 1. The user attempts to access the Recycling Value Calculator without reaching the required 40 points. 2. The user unable to click on the selected activity as it has not been unlocked yet. 3. The user is encouraged to scan waste and start earning points.

Table 3.17 shows the use case description for “Add Feedback” functionality on how it allows user to provide feedback or suggestion regarding the app.

Table 3.17: Use Case Description for Add Feedback

Use Case	Add Feedback
Description	Allow the user to provide feedback on the app.
Actors	User
Preconditions	• The user must have an account. • The user must be logged in.
Postconditions	• Upon successful feedback submission, the feedback is submitted and stored in the system. • Upon unsuccessful feedback submission, an error message is displayed and the feedback is not submitted.
Main Flow	1. The user navigates to the feedback page. 2. The user provides the required input to the required field and submits. 3. The system validates and saves the feedback in the database. 4. A success notification is displayed.
Alternate Flow	(A1) Missing input data: 1. The user provides empty input data. 2. The system displays certain error messages for the empty fields. 3. The user is prompted to provide input to the required field. 4. The user fixes the issue and proceeds with adding feedback process.
Exception Flow	(E1) User cancels adding feedback: At any point during the process, the user may choose to cancel and exit.

Table 3.18 shows the use case description for “View Feedback” functionality on how it allows admin to view the feedback provided by users.

Table 3.18: Use Case Description for View Feedback

Use Case	View Feedback
Description	Allow the admin to view feedback provided by user.
Actors	Admin
Preconditions	• The admin must be logged in. • Feedback data has been submitted by user and exists in the database.
Postconditions	The admin can view the list of feedbacks submitted by users.
Main Flow	1. The admin navigates to feedback management. 2. The system retrieves and displays all feedback. 3. The user can view a list of submitted feedback.
Alternate Flow	(A1) Filter feedback: 1. The admin wants to filter the feedback list by date range, rating, or other criteria. 2. The admin selects the filter options and applies them. 3. The system updates the feedback list based on the selected filter options. (A2) Sort feedback: 1. The admin wants to sort the feedback list by date, rating, or other parameters. 2. The admin selects the sort options and applies them. 3. The system updates the feedback list based on the selected sort options.
Exception Flow	(E1) No feedback available: 1. The admin navigates to feedback management. 2. No feedback items are displayed indicating that no feedback has been added.

Table 3.19 shows the use case description for “Manage User Accounts” functionality on how it allows admin to manage user accounts.

Table 3.19: Use Case Description for Manage User Accounts

Use Case	Manage User Accounts
Description	Allow the admin to manage user accounts.
Actors	Admin
Preconditions	• The admin must be logged in. • User data exists in the database.
Postconditions	• The admin can view the list of registered user accounts. • The user data is updated in the database.
Main Flow	1. The admin navigates to user management. 2. The system retrieves and displays all user accounts. 3. The admin can choose to: a. view a list of registered user accounts. b. change account status (e.g., activate, suspend, ban). c. delete the user account.
Alternate Flow	(A1) Filter user: 1. The admin wants to filter the user list by date joined range, or other criteria. 2. The admin selects the filter options and applies them. 3. The system updates the user list based on the selected filter options. (A2) Sort user: 1. The admin wants to sort the user list by date joined, or other parameters. 2. The admin selects the sort options and applies them. 3. The system updates the user list based on the selected sort options. (A3) Search user: 1. The admin wants to search the user list by user name, or other parameters and provides them in the required field.

	2. The system updates the user list based on the provided value.
Exception Flow	(E1) No user available: 1. The admin navigates to user management. 2. No user accounts are displayed indicating that no user has been registered.

Table 3.20 shows use case description for “Manage Data” functionality on how it allows admin to manage all data.

Table 3.20: Use Case Description for Manage Data

Use Case	Manage Data
Description	Allow the admin to manage all data.
Actors	Admin
Preconditions	• The admin must be logged in. • Data exists in the database.
Postconditions	The admin can view the list of all data.
Main Flow	1. The admin navigates to management screen. 2. The system retrieves and displays all data. 3. The admin can choose to: a. view data. b. add new data. c. edit data. d. delete data.
Alternate Flow	(A1) Filter waste: 1. The admin wants to filter the list by date range, name or other criteria. 2. The admin selects the filter options and applies them. 3. The system updates the list based on the selected filter options. (A2) Sort waste: 1. The admin wants to sort the list by date, alphabetically, or other parameters.

	2. The admin selects the sort options and applies them. 3. The system updates the list based on the selected sort options. (A3) Search waste: 1. The admin wants to search the list by name, or other parameters and provides them in the required field. 2. The system updates the list based on the provided value.
Exception Flow	(E1) No data available: 1. The admin navigates to management. 2. No items are displayed indicating that no data have been added.

Table 3.21 shows the use case description for "Logout" functionality on how it allows users and admin to log out of the app.

Table 3.21: Use Case Description for Logout

Use Case	Logout
Description	Allow the users/admin to log out of the app.
Actors	User, Admin
Preconditions	• The user must have an account. • The user/admin must be currently logged in.
Postconditions	• Upon successful logout, the user's/admin's session is terminated and no longer has an access to the exclusive functionalities. • Upon unsuccessful logout, the user's/admin's remain logged in.
Main Flow	1. The user/admin clicks the logout button. 2. The system prompts the user/admin with a confirmation message. 3. The user/admin confirms the logging out.

	4. The system ends the session and redirects to the home screen.
Alternate Flow	N/A
Exception Flow	(E1) User/Admin cancels logout: At any point during the logout process, the user/admin may choose to cancel and exit the logout flow.

3.4.2.2 Activity Diagram

Figure 3.15 illustrates the flow of activities or processes within the app by showing how the progress of tasks from start to finish.

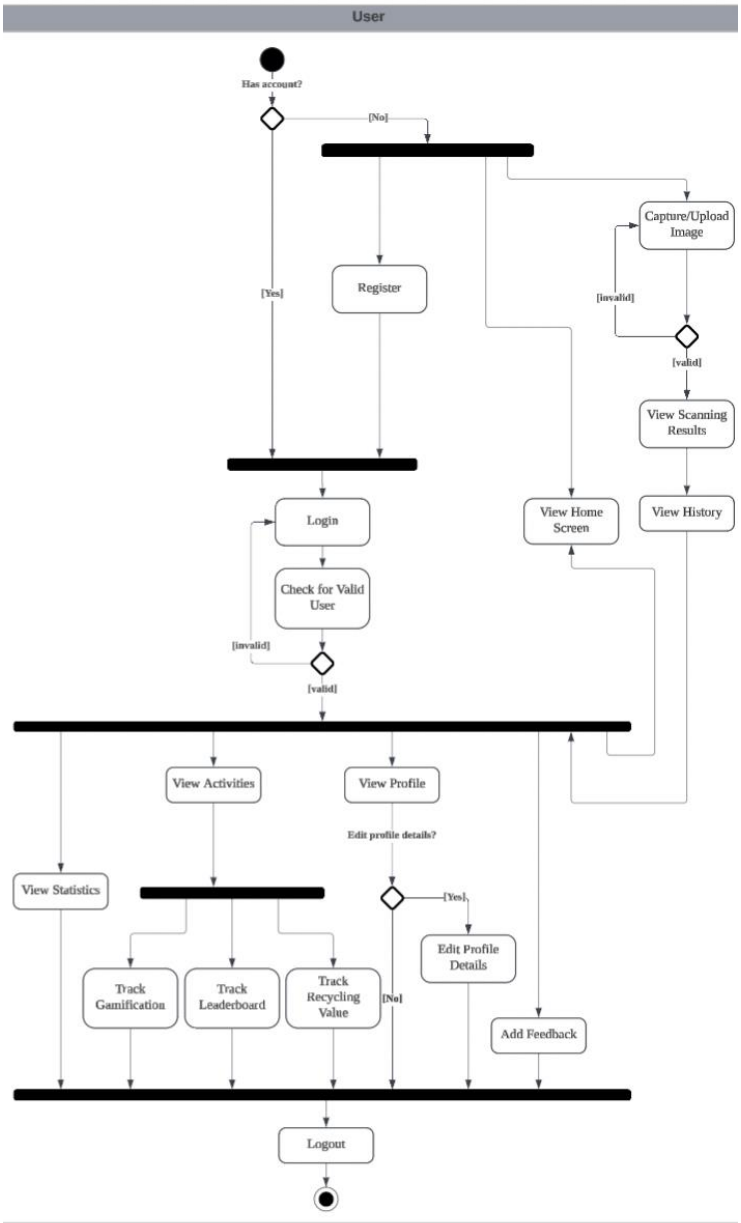


Figure 3.15: Class Diagram

Users without an account are limited to viewing the Home Screen, capturing or uploading images for waste sorting, View Scanning Results, and reviewing their activity history only. Users must register for an account and log in to access the full range of features. The registration process involves creating an account with an email

address or through a third-party service, Google Login. Once logged in, users can access other exclusive features, including managing their profile, viewing waste activities like play gamification, calculate their recycling value, and take a look at the leaderboard. Other than that, it includes viewing detailed statistics, and submitting feedback.

3.4.2.3 Sequence Diagram

Figure 3.16 to Figure 3.25 illustrates the order of interactions between different components or actors in the app by detailing how features are executed step-by-step.

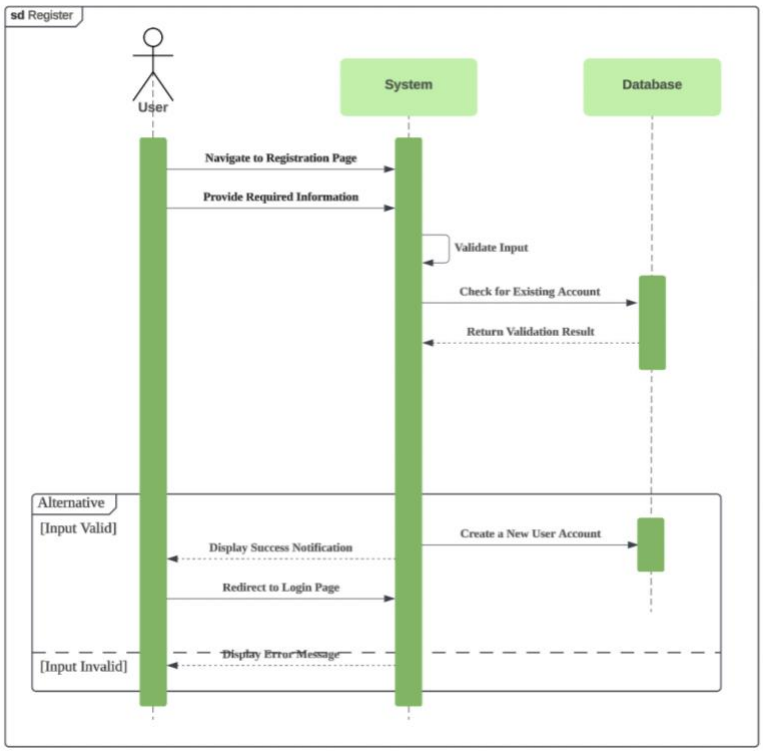


Figure 3.16: Sequence Diagram of Register

Figure 3.16 illustrates the process of user registration to create an account. The user initiates the process by navigating to the registration page and providing the required information. The system validates the input together with checking for

existing accounts. If the input is valid and no existing account is found, a new user account is created in the database, and a success notification is displayed before redirecting the user to the login page. On the other hand, if the input is invalid or there is an existing account with the same email, an error message is displayed.

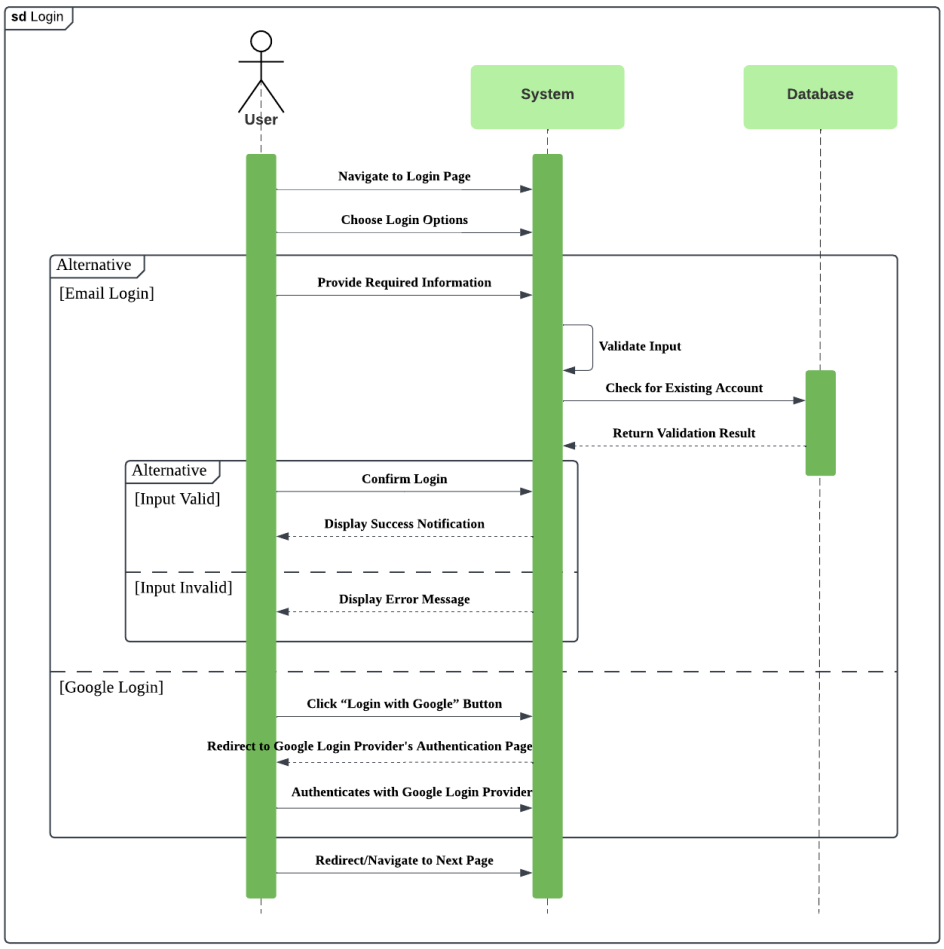


Figure 3.17: Sequence Diagram of Login

Figure 3.17 illustrates the user login process for the registered user. The user navigates to the login page and chooses between email login or Google login. For email login, the user provides the required information, which is then validated by the system. The system checks the database for an existing account. If the input is valid and the account exists, the system confirms the login and displays a success notification. If the input is invalid, an error message is displayed. For Google login, the user clicks the "Login with Google" button, which redirects them to the Google

login provider's authentication page. After successful authentication with Google, the user is redirected to the next page.

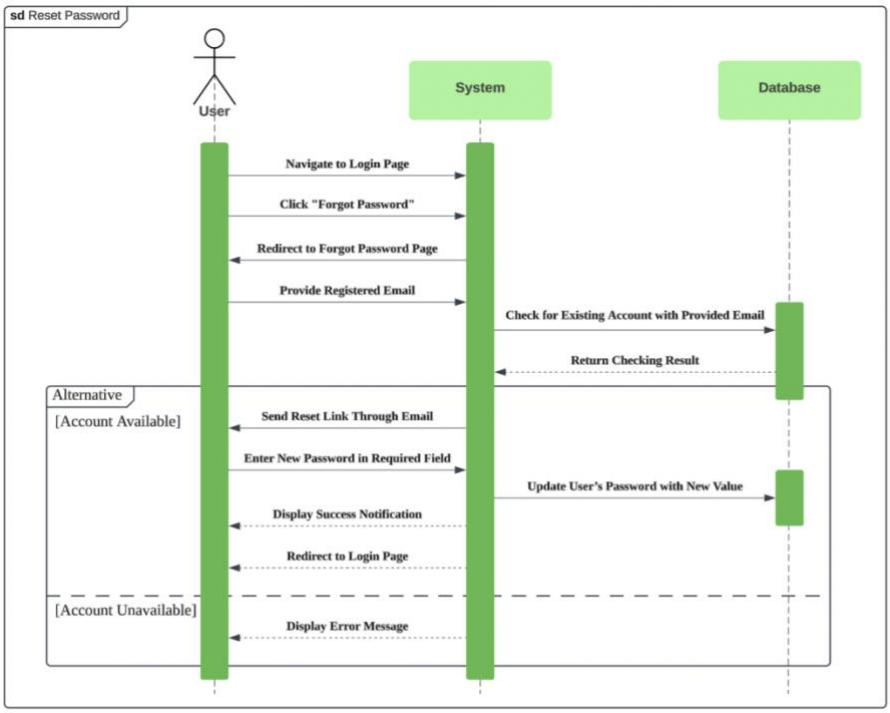


Figure 3.18: Sequence Diagram of Reset Password

Figure 3.18 illustrates the process of resetting a forgotten password. The user initiates the process by clicking "Forgot Password" located in the login page. The user is required to enter their email and immediately provide a new password into the new password input field. It checks the database for an account associated with the provided email. If found, a reset link is sent to the email for verification. Upon successful new password entry, the system updates the user's password in the database and redirects them to the login page. Otherwise, an error message is displayed.

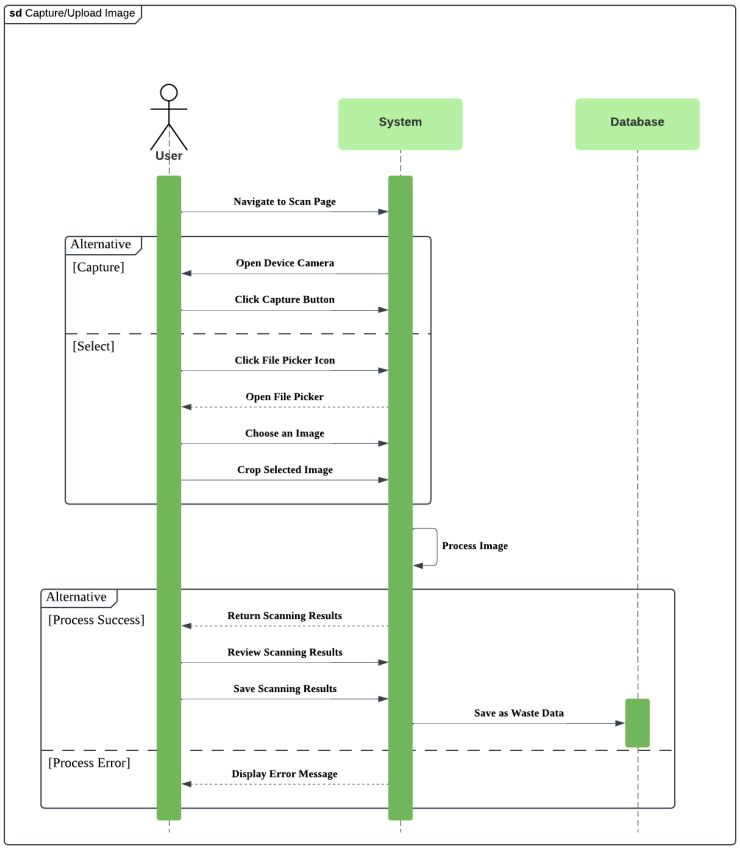


Figure 3.19: Sequence Diagram of Capture/Upload Image and Scanning Process

Figure 3.19 illustrates the process of capturing and then uploading an image for waste sorting. The user navigates to the Scan Page. They then have the option to either capture an image directly from their device camera or select an image from their device's storage. If the user chooses to capture an image, they click the Capture Button to capture an image. If they choose to select an image, they click the File Picker Icon, and the File Picker opens. The user then chooses an image from their device. The selected image can be cropped. The system then processes the image. If the process is successful, the system returns the scanning results, which the user can review and save. If the process fails, an error message is displayed.

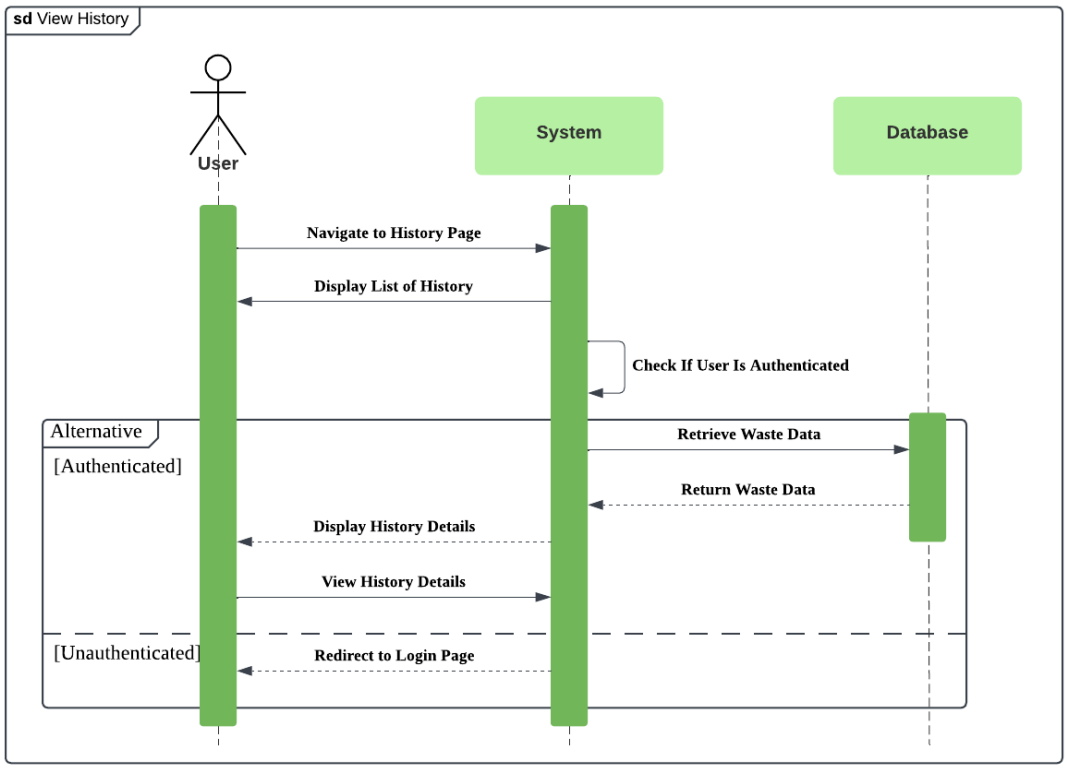


Figure 3.20: Sequence Diagram of View History

Figure 3.20 illustrates the process of viewing waste history. The user navigates to the History Page and requests to view their history. The system checks if the user is authenticated. If the user is authenticated which means that they have an account and logged in, the system retrieves the waste data from the database and returns it to the user. The user can then view the history details. If the user is not authenticated, the system redirects them to the login page for them to log in or create an account.

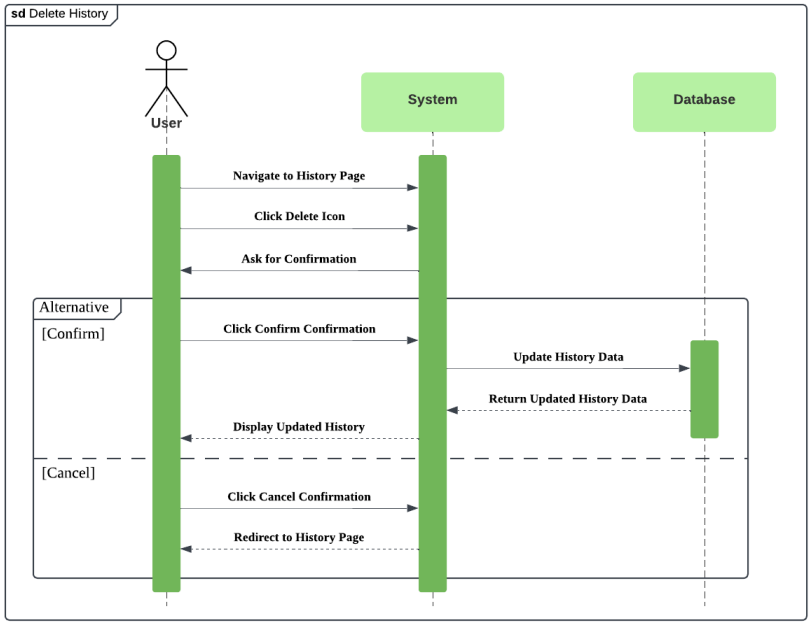


Figure 3.21: Sequence Diagram of Delete History

Figure 3.21 illustrates the process of deleting a history entry. The user navigates to the history page and clicks the delete icon next to the desired item. The system then asks for confirmation from the user. If the user confirms the deletion, the system updates the history data in the database and returns the updated history data to the user. The user can then view the updated history. If the user cancels the deletion, the system redirects them back to history page without making any changes.

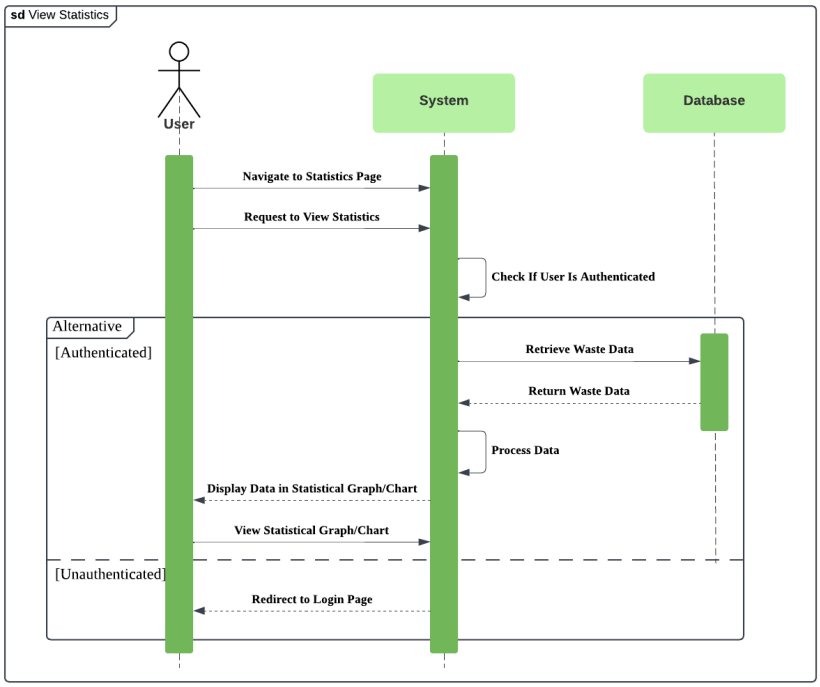


Figure 3.22: Sequence Diagram of View Statistics

Figure 3.22 illustrates the process of viewing waste statistics. The user navigates to the statistics page and requests to view the statistics. The system checks if the user is authenticated. If the user is authenticated which means that they have an account and logged in, the system retrieves the waste data from the database and returns it to the user. The user can then view the data displayed in a statistical graph or chart. If the user is not authenticated, the system redirects them to the login page for them to log in or create an account.

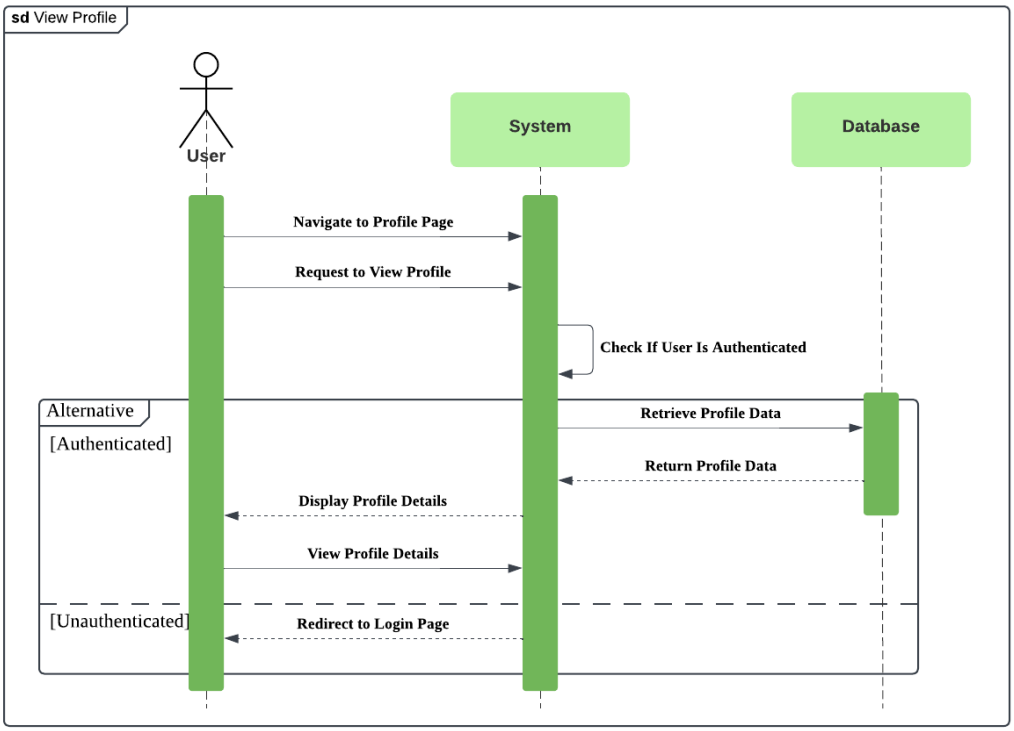


Figure 3.23: Sequence Diagram of View Profile

Figure 3.23 illustrates the process of viewing the user profile. The user navigates to the profile page and requests to view their profile. The system checks if the user is authenticated. If the user is authenticated which means that they have an account and logged in, the system retrieves the profile data from the database and returns it to the user. The user can then view their profile details. If the user is not authenticated, the system redirects them to the login page for them to log in or create an account.

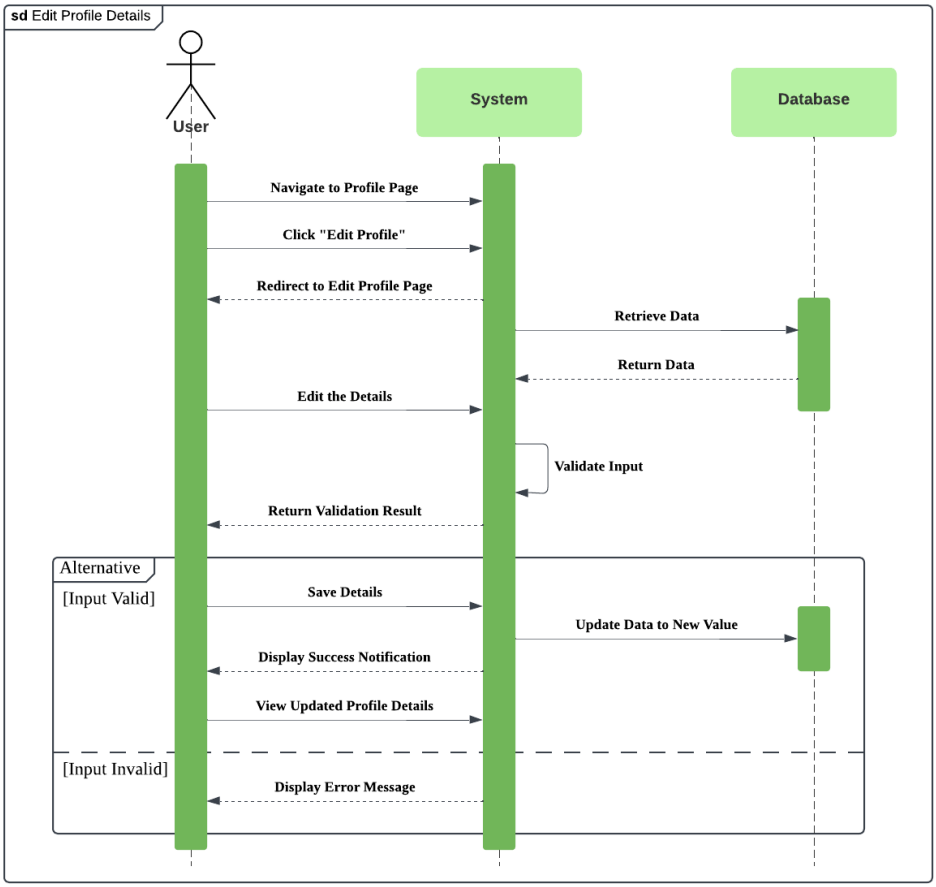


Figure 3.24: Sequence Diagram of Edit Profile Details

Figure 3.24 illustrates the process of editing profile details. The user navigates to the profile page and clicks “Edit Profile”. The system redirects the user to the edit profile page and retrieves the existing data from the database. The user edits the details and returns the changes to the system. The system validates the input and returns the validation result. If the input is valid, the system updates the data in the database with the new values, displays a success notification, and allows the user to view the updated profile details. If the input is invalid, the system displays an error message. This process does not have authentication as it has already been completed in the view profile process.

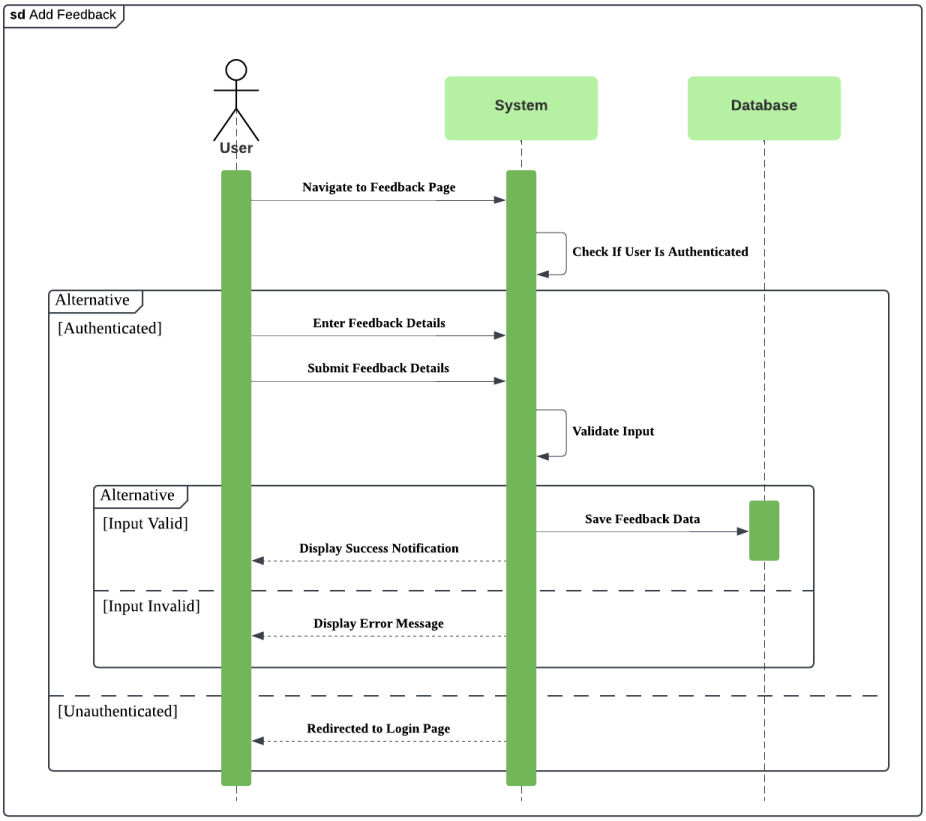


Figure 3.25: Sequence Diagram of Add Feedback

Figure 3.25 illustrates the process of adding feedback within a system. The user navigates to the feedback page. The system then checks if the user is authenticated. If the user is authenticated which means that they have an account and logged in, they can enter their feedback details and submit them. The system validates the input. If the input is valid, the feedback data is saved to the database, and a success notification is displayed. If the input is invalid, an error message is displayed. If the user is not authenticated, they are redirected to the login page for them to log in or create an account.

3.4.2.4 Class Diagram

Figure 3.26 illustrates the structure of the proposed application's data, including the classes, attributes, and relationships that define its core components.

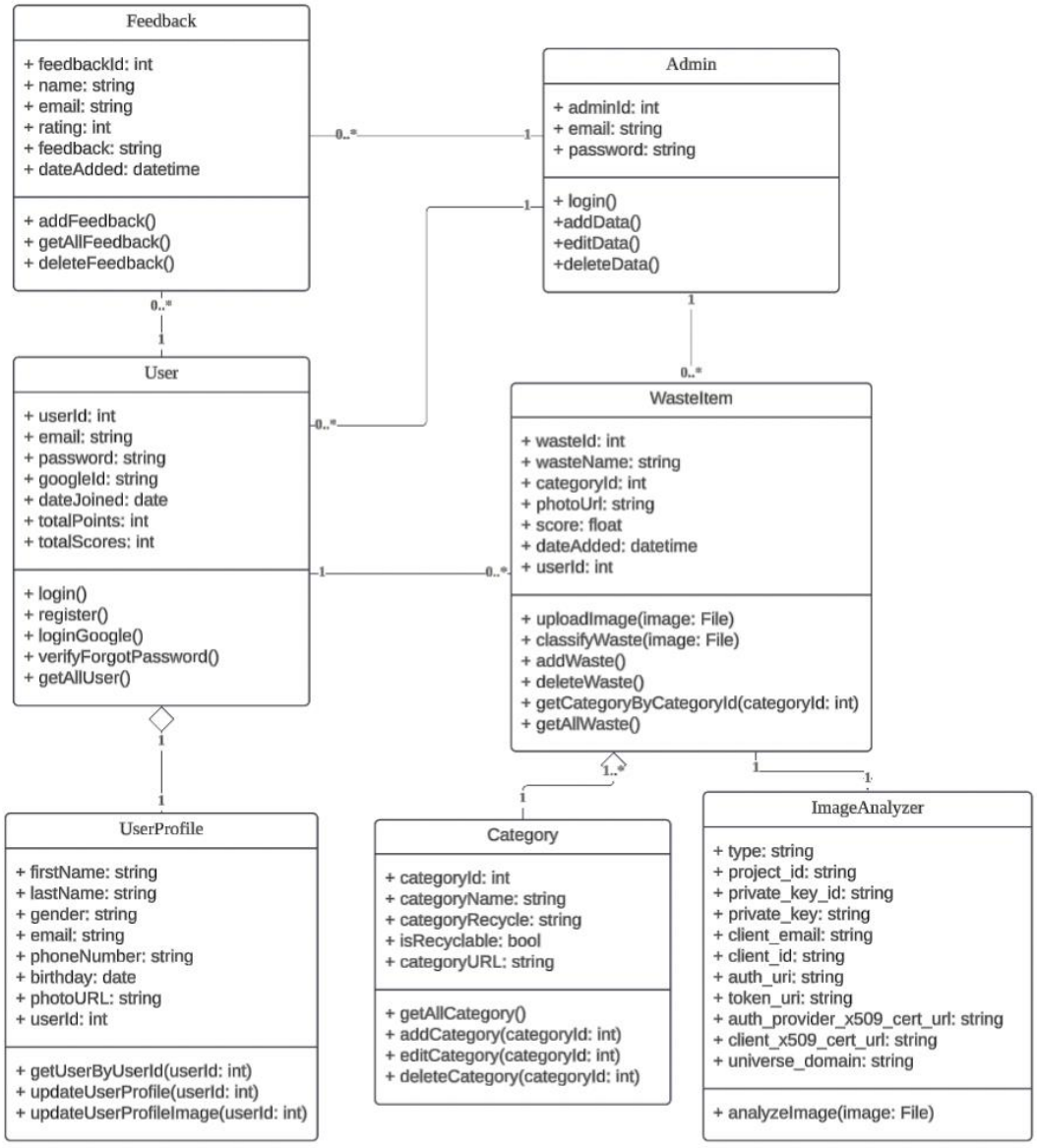


Figure 3.26: Class Diagram

The User class has a one-to-one relationship with the UserProfile class which means a user can have one user profile. The Admin class has a one-to-many relationship with the User class which means an admin can manage multiple users. The User class has a one-to-many relationship with the Feedback class which means

a user can submit multiple feedback entries. The WasteItem class has a one-to-many relationship with the Category class which means a category can belong to multiple waste items. The Admin class has a one-to-many relationship with the Feedback class which means an admin can view multiple feedback.

3.4.3 Physical Design

Figure 3.27: Home Screen

Figure 3.27 shows the home screen which is the first screen the user will interact with as soon as they open the app. If the user is logged in, it will show the points, days joined, and the number of items sorted.

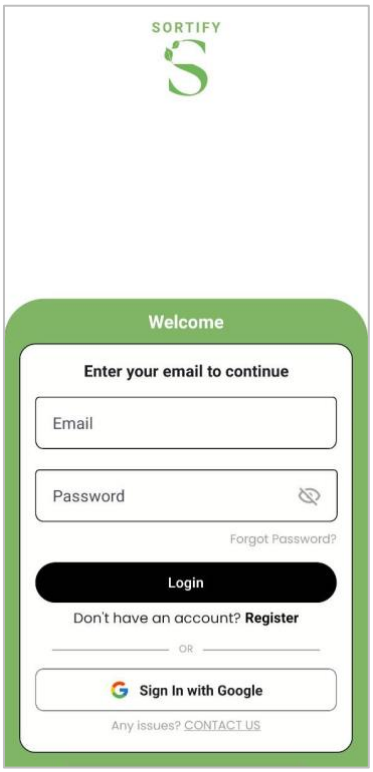


Figure 3.28: Login Screen

Figure 3.28 shows the login screen where the user needs to enter their email and password to log into the app. They can also login with Google. By clicking "Register", it will navigate the user to the register screen as shown in Figure 3.29.

Figure 3.29: Register Screen

Figure 3.29 shows the register screen where the user needs to provide their name, email and password to create an account.

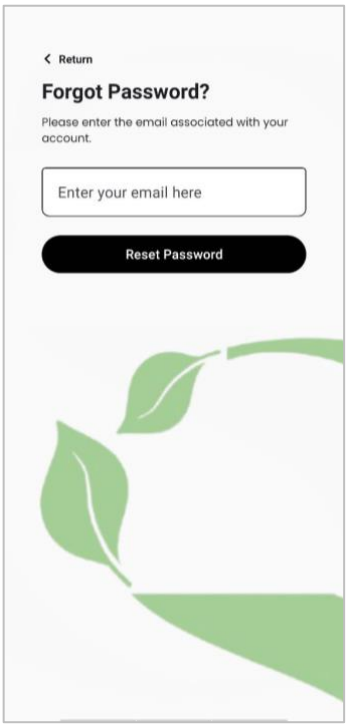


Figure 3.30: Forgot Password Screen

Figure 3.30 shows the forgot password screen where the user needs to enter their email that associated with existing account. After submitting, the system will send a reset link through email and user can set their new password.

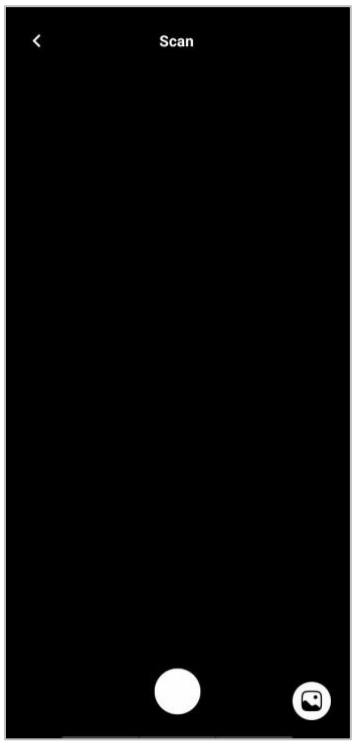


Figure 3.31: Scanner Screen

Figure 3.31 shows the scanner screen which is main feature of the app. The user can directly capture the image by clicking the capture button or can click the file picker icon at bottom right to open the gallery.

Figure 3.32: Scanning Results Screen

Figure 3.32 shows the scanning results screen where a box will appear from the bottom. The box consists of the scanning results, save button, and report button.

Figure 3.33: List of Activities Screen

Figure 3.33 shows a screen of the list of activities. The activities will be unlocked once the user reach certain tier.

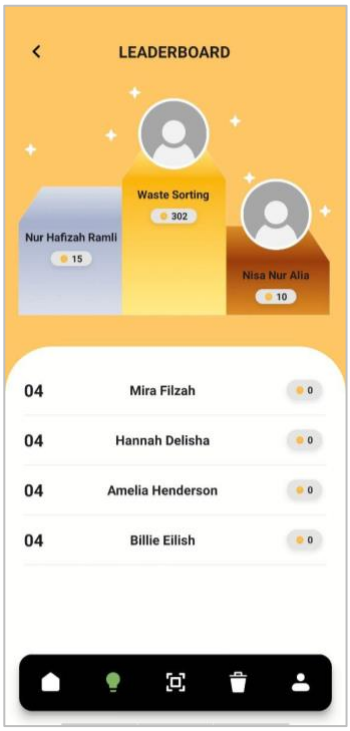


Figure 3.34: Leaderboard Screen

Figure 3.34 shows the leaderboard where the ranks are decided by total of points collected by the users. The user with the highest total of point will take the first place and so on.

Figure 3.35: Gamification Screen

Figure 3.35 shows the game, “Know Your Trash”, where firstly the game instructions will display. User can click “Start Game” to start the game and collect scores.

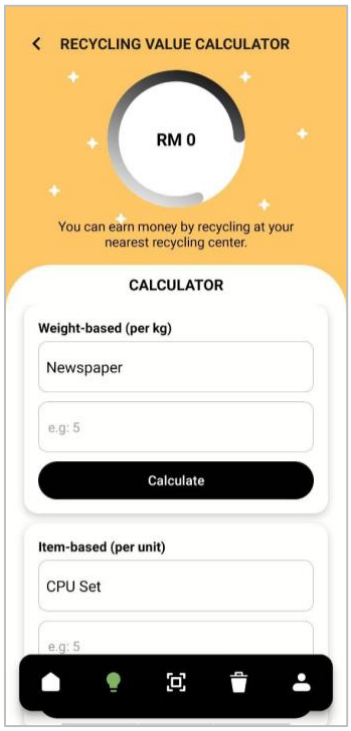


Figure 3.36: Recycling Value Calculator

Figure 3.36 shows the recycling value where the top section is the total monetary value of the user's waste collection based on the current market rates. The bottom section states the categories that contributes to the monetary value.

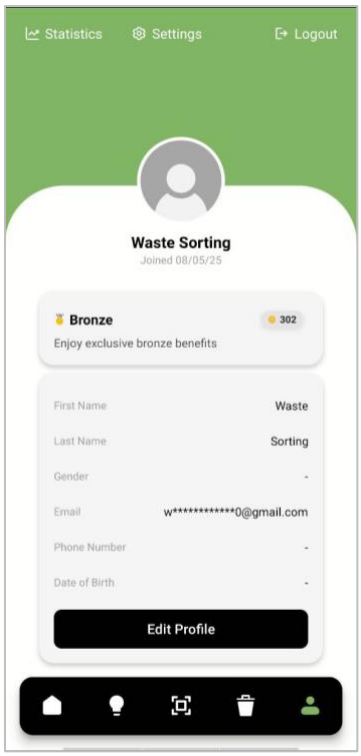


Figure 3.37: Profile Screen

Figure 3.37 shows the profile screen that includes the statistics button that will lead to the statistics page as shown in Figure 3.39, settings button, logout button, profile image, member tier for registered user, and a glimpse of the profile details. By clicking the "Edit Profile" button, it will navigate the user to the page as shown in Figure 3.38.

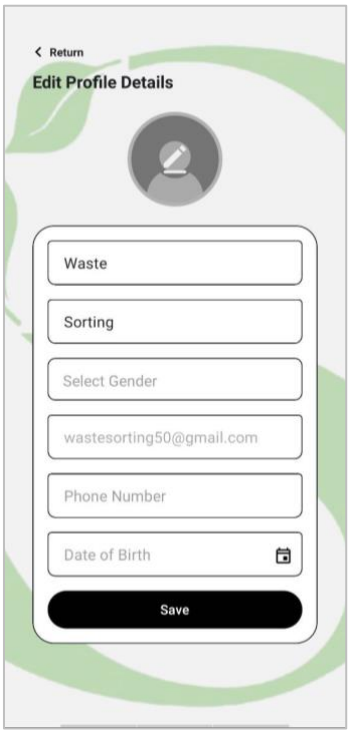


Figure 3.38: Edit Profile Screen

Figure 3.38 shows the edit profile screen which consist of the form that includes first name, last name, gender, email, phone number, and date of birth fields.

Figure 3.39: Statistics Screen

Figure 3.39 shows the statistics screen where the top section shows the statistical graph of total waste items scanned. The bottom section is a pie chart of the waste types recognized.

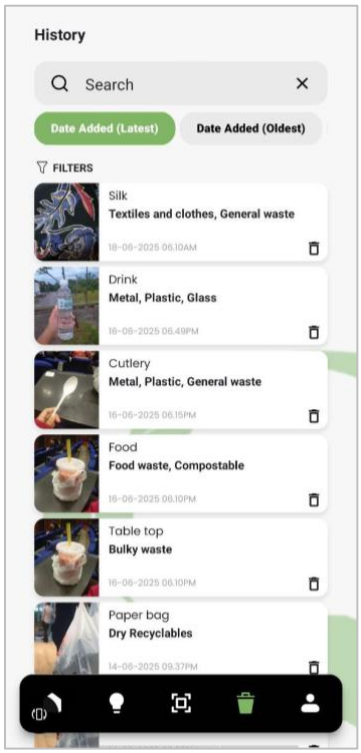


Figure 3.40: History Screen

Figure 3.40 shows the history screen where a list of scanned waste items will be listed, and it is scrollable. This page consists of search bar, filter button, and sort dropdown where a container will open as soon as the user click the filter button.

3.5 Chapter Summary

This chapter focuses on the important steps of requirement analysis and design for he proposed application. It emphasizes the methodology of the proposed application with utilizes Agile approach that offers advantages for the proposed application particularly in term of efficient time management, cost reduction and iterative development. The analysis phase looks at data, the system itself, and the functions it needs to perform, helping to clarify how the system will work and meet user expectations. The design phase discusses the system's structure, how it will be logically organized, and the design of the user interfaces, giving a clear plan for building the solution.

CHAPTER 4: IMPLEMENTATION AND TESTING

4.1 Introduction

This chapter presents the detailed implementation and testing phases of the proposed application. After determining the problem and architectural design of the system in the previous chapters, the focus now shifts to how the system was developed and tested. This chapter discusses the selected technologies, their roles in achieving the system's objectives, and explaining each of the main modules of the application. In addition, this chapter discovers the execution flow from the user's point of view and the administrator that provides an overview of the project's code structure, and describes the main challenges together with its solution. The development process has followed a modular and iterative approach, with continuous testing and improvements based on feedback and system requirements. A big part of the testing focused on the integration of the Google Vision API, which enables the object recognition and label detection features in the app.

4.2 Implementation

The implementation phase focuses on converting the planned system design into a working application. This involves setting up the front-end and back-end, integrating third-party services, and ensuring that each component works together as intended.

4.2.1 Technologies Used

The development required the use of several key technologies to ensure that the system functions effectively and is user-friendly. Each tool and platform was chosen based on its suitability to meet the project's objectives.

4.2.1.1 React Native

React Native, an open-source mobile application framework developed by Meta (Joshi, 2025). In building this mobile application, React Native has been selected from other frameworks such as Flutter because of its ability to build cross-platform apps (Android and iOS) by only using a single JavaScript codebase. In addition, previously experiences with React for web development reduces the learning curves for React Native. This facilitates faster development and has allowed for greater code consistency and maintainability across both web and mobile platforms. Moreover, one of its main advantages is that React Native also allows the reuse of codes, components or logic developed from previous React projects that can be effectively adjusted for this proposed project. React Native has a broader community and resource which makes it more accessible and easier to learn compared to other frameworks. Additionally, since the app frequently interacts with native device features like the camera for image scanning, React Native is a practical choice due to its robust integration with native modules.

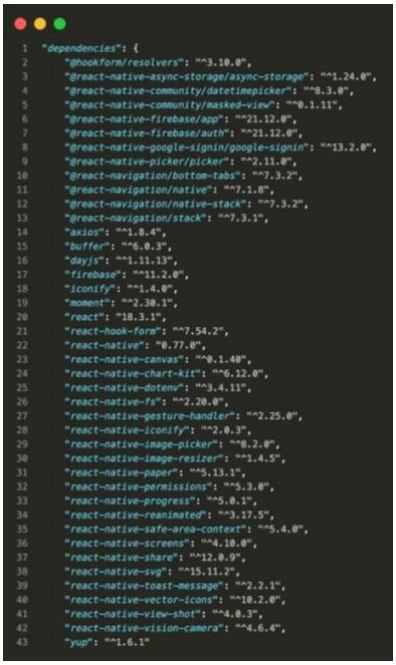


Figure 4.1: List of Project Dependencies

Figure 4.1 shows a range of dependencies and packages to improve the development efficiency, support essential functionalities, and improve the overall user experience. These dependencies include libraries for the user interface components, @react-navigation for seamless navigation, form handling with react-hook-form, network requests using axios, @react-native-firebase and @react-native-google-signin for authentication and real-time data synchronization. Then, several dependencies such as react-native-image-picker, react-native-image-resizer, react-native-vision-camera, and react-native-fs are used to enable the camera functionalities which are for capturing and processing images during the scanning process. These dependencies ensure that image input from users can be used effectively to identify and classify waste.

4.2.1.2 Firebase

Firebase is one of the most widely used technologies on the market today that provide a range of features such as real-time data synchronization, user authentication, cloud storage, and static hosting, all of which are supported by Google (Sureka, 2022). This project used several Firebase services such as Firebase Authentication that manages the registration and login processes, Cloud Firestore, Firebase Storage, and Firebase Rules to protect sensitive data.

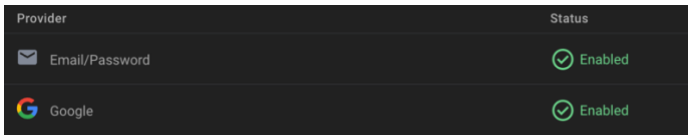


Figure 4.2: Firebase Authentication

Two authentication providers are enabled which are email/password and Google Sign In as shown in Figure 4.2. Cloud Firestore, a NoSQL cloud database, has a real-time syncing and scalability that provide a simple way to store data. By

using Firebase, it simplified the backend deployment and reduced the need to manage a dedicated server or database manually.

4.2.1.3 Google Vision API

Google Vision API is a cloud-based machine learning service that extracts information from images and was integrated in this project to support image-based waste classification. For this project, two specific features were integrated which are object localization and label detection. Both were chosen because certain objects may be detected by one feature but not the other one that depends on the image angles or background complexity.

Label detection provides general labels that describe the overall content of the image such as items, places, activities, animals, or other types of elements. Meanwhile, object localization focuses more detailed analysis by detecting and locating specific objects in the image using bounding boxes (Daminion, 2023). These bounding boxes help visually highlight the identified objects in the image, which is useful in cluttered or multi-object scenarios. By combining object localization and label detection, the system ensures more robust, accurate, and user-friendly classification, even in cases where the image is not ideal.

While Google does not explicitly state which machine learning model powers the Vision API, it is reasonable to assume that EfficientNet, a convolutional neural network (CNN) developed by Google, is part of its underlying architecture. EfficientNet has demonstrated superior performance in various image classification tasks due to its balanced scaling of depth, width, and resolution (Tan & Le, 2019). Furthermore, article such as “DeepFake Image Detection and Classification using EfficientNet Model” by Singh et al. have shown that EfficientNet performs

exceptionally well in high-accuracy classification scenarios like waste image recognition. Given this, it is plausible that Google Vision API utilizes EfficientNet or its variants to achieve its accurate and efficient image labelling and object detection capabilities.

In terms of cost management, the application is designed to operate within the Google Cloud free tier, which includes 1,000 units per month for Vision API usage at no charge. Usage is carefully monitored before sending to stay within these limits and prevent unexpected charges. If usage exceeds the free tier, additional costs will be covered, with pricing starting at approximately RM 6.55 per 1,000 units for Label Detection and RM 9.82 per 1,000 units for Object Localization.

4.2.1.4 Dataset Used

A publicly available dataset was used from Kaggle titled “Waste Classification Data” to support the classification logic and expand the application’s object list. The dataset contains 22500 images of organic and recyclable waste items (Sekar, 2019) and approximately 1000 images were selected for testing purposes. In addition, 940 images sourced from Google were also used for testing. These images were categorized into four types which are single item with a white background, single item with a dark background, multiple items with a white background, and multiple items with a dark background. While the app does not train a machine learning model directly, the dataset was applied during testing to have a better understand of typical waste item labels and to build a structured object-category mapping in the Firebase database.

4.2.1.5 Other Tools

In addition to the core technologies, there are several other tools that have supported the implementation process. Visual Studio Code, a powerful editor, plays a big role in developing this project as it provides an effective environment for writing, organizing, and maintaining the codebase. Next, Android Studio is primarily used for its Device Manager feature as it allows an emulation of the mobile application on different Android devices to ensure that the application can work properly across screen sizes and configurations for an efficient testing. In addition, Figma is used during the design phase to create the UI/UX mockups, layout flows, and visual components of the application before its implementation. Finally, Git is used to ensure a track of code changes for code reviews to prevent data loss during development. Figure 4.3 shows the repository for this project in GitHub. These tools have been chosen because of their reliability, strong community support, and ability to integrate smoothly into the project's workflow.

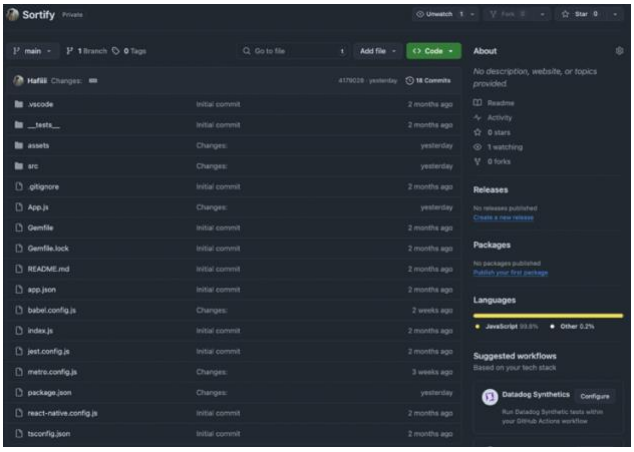


Figure 4.3: Sortify Repository in GitHub

4.2.2 System Modules

This section explains the main modules that make up the application. These modules work together to handle image input, communicate with the Google Vision API, display results, manage waste item data, and support administrative actions.

4.2.2.1 Front-End Implementation

This section refers to the user interface and experience (UI/UX) part of the app, which is implemented entirely in React Native. The structure is organized under the "screens" directory, where each folder corresponds to a specific screen or functional section of the mobile app. This modular design helps in maintaining scalability, readability, and ease of navigation. Figure 4.4 shows some of the main modules.

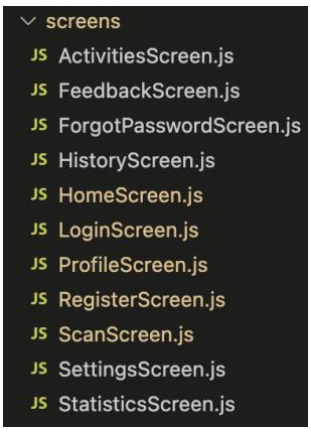


Figure 4.4: Files in Screen Folder

"ActivitiesScreen" displays a list of some fun activities or games related to waste sorting or environmental conservation. "FeedbackScreen" allows users to submit their opinions, suggestions, or report issues within the app for the developers to improve the app experience. "ForgotPasswordScreen" handles the password recovery process. "HistoryScreen" presents a chronological list of all scanned waste items by the user to keep track of their personal disposal habits. "HomeScreen" acts as the main landing page. "LoginScreen" acts as the entry point for returning users

and admin to access their personalized content by logging in using their email and password or through Google Sign In. “ProfileScreen” shows the user’s profile details and allows them to edit. “RegisterScreen” allows new users to create an account by providing necessary information. “ScanScreen” is the core functionality of the app, enabling users to capture or upload an image of waste, which then processed via the Google Vision API to classify the waste type and provide disposal guidance. “SettingsScreen” offers app preferences and customization options. “StatisticsScreen” visualizes waste management data through charts and summaries to motivate improvement through data-driven insights. Each screen is styled using react-native-paper and responsive design using Flexbox and ScrollView to support different screen sizes.

4.2.2.2 Backend Integration

The backend of the Waste Sorting Application was developed using Firebase Firestore and Firebase Storage. Both offer real-time performance, scalability, and ease of integration with React Native. Firestore serves as the main cloud database and Firebase Storage is used to handle images uploaded by user. In this project, the database consists of multiple collections such as “categories”, “counters”, “feedbacks”, “issues”, “objects”, “users”, and “wastes” as shown in Figure 4.5.

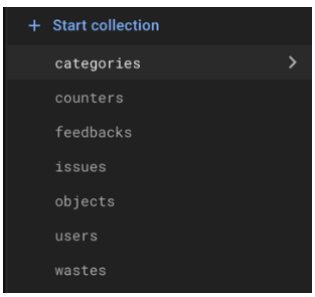


Figure 4.5: Firestore Collections

The “users” collection stores user information, including first name, last name, email address, phone number, profile photo URL, gender, birthday, date joined, and a unique user ID. The “counters” collection is used to increment and track user IDs each time a new user registers. The “categories” collection contains predefined waste categories, each with a category ID, name, image URL, description, and a Boolean field indicating whether the category is recyclable. The “objects” collection contains information about common item names that can be recognized by the system and are linked to a corresponding “categoryId” from the “categories” collection. The “wastes” collection saves scanned waste records, including the waste name, detection results such as waste type and confidence score, image URL, date added, and associated user ID. The “feedback” and “issues” collections record the feedback and issues submitted and reported by user during app usage.

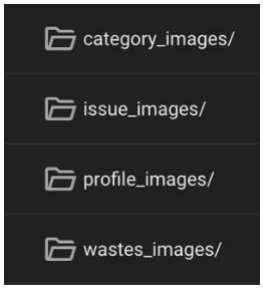


Figure 4.6: Firebase Storage Files

Each time a user scans an item, submits an issue, updates their profile image, or when an admin uploads a category image, the image is first validated and then uploaded to Firebase Storage. Once uploaded, Firebase returns a public URL, which is then stored in Firestore under the relevant collection. Figure 4.6 shows the list of files in the Firebase Storage which are “category_images”, “issue_image”m “profile_image”, and “wastes_images”. Each directory helps organize the uploaded images based on their purpose to ensure easy management.

The backend also supports administrator actions. Admin users can log in securely to manage entries in the objects and categories collections via the content management system (CMS). This includes adding new object or category, updating information, and reviewing user feedback or reports. Firestore's real-time capabilities ensure that any updates made by the admin are immediately updated in the app without requiring manual refreshes.

4.2.2.3 Google Vision API Integration

Images captured or selected by users are first converted into Base64-encoded strings to perform the detection. These encoded images are then sent in a POST request to the Vision API using Axios. Figure 4.7 shows the snippet that demonstrates the implementation of this request.

```
const response = await axios.post(
            `https://vision.googleapis.com/v1/images:annotate?key=${GOOGLE_VISION_API_KEY}`,
            {
                requests: [
                    {
                        image: { content: base64Image },
                        features: [
                            { type: "OBJECT_LOCALIZATION", maxResults: 10 },
                            { type: "LABEL_DETECTION", maxResults: 10 }
                        ]
                    }
                ]
            }
        );
```

Figure 4.7: Vision API request

The API responds with a list of label annotations and localized objects together with its confidence score. The application processes both sets of results and removes any duplicate labels. The unique list of detected objects is then matched against the existing entries in the Firestore "objects" collection. If a match is found, the corresponding "categoryId" is retrieved and used to access additional details from the categories collection, such as disposal instructions and recyclability status. Once the classification is completed, the application uploads the image to Firebase Storage, and saves the image URL and classification results to the "wastes"

collection in Firestore. This enables users to maintain a personalized history of scanned waste items for future reference.

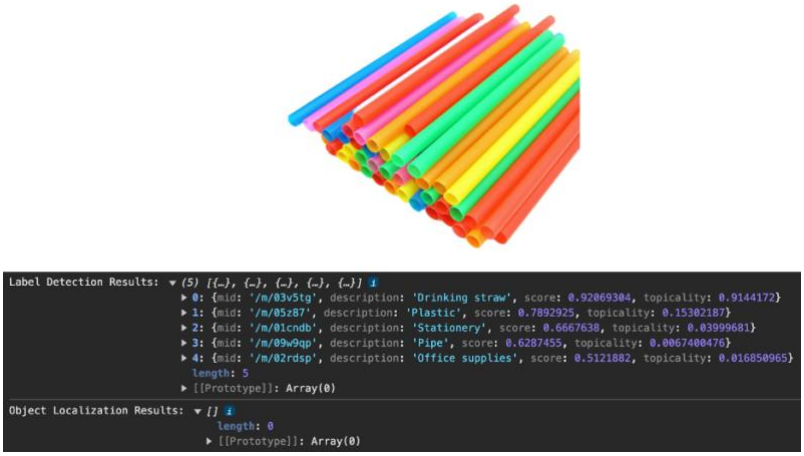


Figure 4.8: Label Detection and Object Localization Results

An example is shown in Figure 4.8. An image of plastic straws returns a label detection result of "Drinking Straw" with the highest confidence score, while the object localization feature fails to detect a defined object. This scenario is common when objects lack clear edges, is photographed at an awkward angle, or if the API lacks training data for that specific context. Even in such cases, label detection provides sufficient information for the app to determine the correct waste category. This proves that it is important to integrate both features, label detection and object localization, into the system to get accurate identification and classification of wastes.

4.2.3 Flow of Execution

This section outlines the step-by-step process of user interactions, image processing, and communication with the Google Vision API. Additionally, it explains how the system handles different user roles, such as regular users and administrators.

4.2.3.1 Regular User Workflow

For regular users, after authentication, the app directs them to the main user interface composed of a bottom tab navigator with several key sections which are Home, Activities, Scan, History, and Profile as shown in Figure 4.9. Customer actions are smoothly managed through stack navigators for logical screen grouping and seamless navigation.

Figure 4.9: Bottom Bar

Each registered user is assigned with a unique “userId” greater than 5 to differentiate them from administrative accounts in the system. This “userId” is used throughout the application to link personal data, such as waste scan history, feedback submissions, and activity logs. For unregistered users or users who choose to explore the app without signing in, features access is limited. They can only view the Home and Scan screens. This limited access still allows them to scan waste items and receive instant classification results using the Google Vision API. However, in order to save scans to track waste sorting history, they are prompted to create an account or log in. This approach ensures the app remains usable and helpful even to new or casual users, while encouraging registration for access to more advanced features and long-term benefits.

4.2.3.2 Administrator Workflow

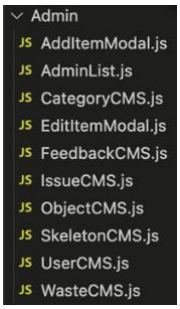


Figure 4.10: Files in Admin Folder

Administrators have access to specialized Content Management System (CMS) screens for managing and maintaining the overall system. Administrators are identified in the system by having a "userId" less than or equal to 5 to differentiate from regular users. Figure 4.10 shows the "Admin" folder which is to support the administrative tasks by organizing the admin interface into some modules.

"CategoryCMS" manages waste categories where Admins can add, edit, or remove categories to ensure accurate classification and up-to-date disposal guidance. "ObjectCMS" is used to maintain the object reference database. Admins can delete the object data, associate them with waste categories, and add new entries for objects that are not detected by the Google Vision API. This ensures continuous improvement and accuracy of the waste classification. "FeedbackCMS" and "IssueCMS" allows admins to view user feedbacks or issues that are submitted through the application for future improvement. "UserCMS" is to manage user accounts by viewing user profiles, checking registration details, and applying actions such as disabling or promoting users as needed to maintain community standards and system integrity. "WasteCMS" displays all recorded waste entries scanned by users to ensure accurate tagging, track common waste patterns, and audit detection results. The other files are just the components in creating the CMS.

These screens allow admins to perform CRUD operations and manage the app content dynamically. As more users scan waste through the app, the object list naturally expands. Admins are responsible for validating and updating this growing database. If certain objects are not detected by the API, they can be manually added through the "ObjectCMS" module to improve future classification. This separation ensures administrators have a secure, dedicated area to perform high-level management tasks without interference from the general user flows, while maintaining consistent navigation and theming across all app areas.

4.2.4 Code Structure and File Organization

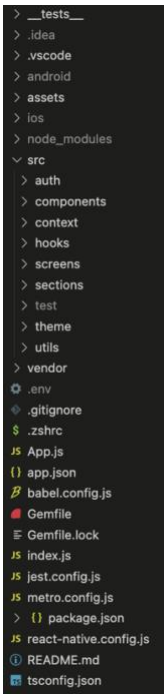


Figure 4.11: Code Files Repository

Figure 4.11 shows the basic structure of the project's source code that was organized into folders. The codebase follows a modular and scalable structure centred around React Native and React Navigation. At the root is the "App.js" file which initializes global providers and configures navigation. The "AuthContext" manages user authentication and global user state, accessed via a custom hook "useAuth". This enables

components to reactively respond to login state changes. The application uses React Navigation's stack and tab navigators to switch between authentication-related screens, the main app interface, and admin panels based on user ID. The app uses react-native-paper for UI components with a custom theme extending the default. The theme is provided globally to maintain consistent colours and fonts. Additional reusable components like Toast notifications and icon sets, Iconify enhance the UI experience without cluttering screen logic.

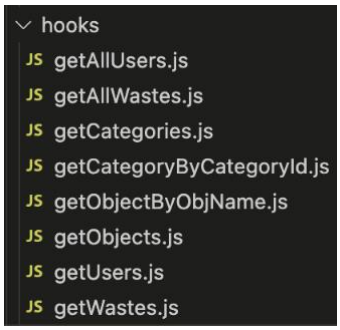


Figure 4.12: Files in Hooks Folder

Figure 4.12 shows the directory in the project that contains custom React hooks responsible for fetching and managing data from the backend API. These hooks abstract away the logic of making API requests and provide reusable, clean interfaces for various components to consume data asynchronously. Each hook corresponds to a specific data entity in the application to ensure maintainable code.

"getAllUsers" fetches a list of all users from the backend. It is used primarily in administrative to display and manipulate user data. "getAllWastes" fetches all waste records stored in the system and it is also useful for administrative reports that require a comprehensive overview of waste items. "getCategories" fetches all waste categories available in the database to associate with the Google Vision API same goes with "getObjects". "getObjects" fetches all waste objects available in the database. "getCategoryByCategoryId" fetches detailed information about a specific waste category based on its unique category ID. It is used when detailed category information is required,

such as in displaying waste category information for users to learn same goes with "getObjectByObjName". "getObjectByObjName" fetches data for a particular waste object by its name. "getUsers" fetches the currently authenticated user's data to personalize the app experience. "getWastes" fetches currently authenticated user's waste data to supports waste tracking, history views, and data analytics. Each hook typically manages its own loading and error states internally.

4.3 Testing

This section outlines the testing efforts carried out to ensure that the application functions as intended and meets the project's objectives. A series of structured test cases were designed and executed to evaluate the system across different criteria, including functionality, reliability, usability, and efficiency.

4.3.1 Testing Strategy and Environment

The testing strategy for this project followed a black-box testing approach, where the system functionalities were evaluated based on inputs and expected outputs without considering the internal code structure. The aim was to verify that the application met all functional and non-functional requirements, particularly for waste scanning using the Google Vision API, user interaction flows, and admin content management.

Functional testing was carried out for major modules, including waste scanning, user authentication, and admin content management. Each feature was tested with valid, invalid, and edge case inputs to evaluate system responses under different scenarios. The application was tested primarily on Android devices, with various screen sizes to ensure responsive design and compatibility. Firebase Firestore and Storage were tested for data

consistency, and Google Vision API was validated using a range of real-world waste images under different lighting conditions and backgrounds.

Informal user testing was conducted with a small group of testers who provided feedback on layout, ease of use, and clarity of features to evaluate the usability. This was supplemented by user surveys that collected subjective evaluations of the interface, accuracy of results, and satisfaction. This testing environment ensured that the app was validated in conditions similar to real-world usage and maintained consistency across devices and networks.

4.3.2 Test Cases and Results

Tables 4.1 to 4.3 present the functionality test cases, while Tables 4.4 to 4.6 cover the reliability tests. Tables 4.7 to 4.9 summarize the tests conducted for efficiency.

Functionality

Table 4.1: Test Case Functionality - Waste Scanning (Google Vision API)

Waste Scanning (Google Vision API)							
ID	Test Scenario	Test Description	Test Data	Expected Result	Status	Severity	Notes
F1	Camera permission granted	Choose “Only this time” or “While using the app”.	N/A	Camera launches successfully.	Pass	High	N/A
F2	Camera permission denied	Choose “Don’t allow”.	N/A	Camera won’t work and only file picker option works.	Pass	Medium	App should not crash if permission is denied.
F3	Capture image	Take photo using camera.	N/A	Image captured and previewed in app.	Pass	High	Ensure phone storage is enough.
F4	Select image from gallery	Tap file picker icon at bottom right of the screen. Choose an image.	N/A	Selected image appears and proceeds to scanning.	Pass	High	N/A

F5	Scan known object	Upload clear image of known item.		Display the item's label with its confidence score and the result.	Pass	High	Label should match object data in Firestore.
F6	Scan unknown object	Upload image not in objects list.		Display "No item detected".	Pass	Medium	N/A
F7	Display scan result	Complete analyzing image process.	N/A	Display label, category, and its description.	Pass	High	N/A
F8	Save scan result	Confirm scan result by pressing "Sorted!" button.	N/A	Data stores in Firestore under "wastes" with user ID.	Pass	High	Able to view in History page.
F9	Report inaccurate result	Tap flag icon beside "Sorted!" button. Submit the report.	N/A	Report saves in Firestore under "issues".	Pass	Medium	Admin can review the report.

Table 4.2: Test Case Functionality - User Authentication

User Authentication						
ID	Test Scenario	Test Description	Expected Result	Status	Severity	Notes
F10	Register new user	Navigate to Register screen. Enter all fields correctly and submit.	Account created in Firebase users collection	Pass	High	N/A

F11	Register with missing fields	Navigate to Register screen. Leave email blank and submit.	Display validation error and unable to submit.	Pass	Medium	N/A
F12	Register with invalid email	Navigate to Register screen. Enter invalid email format and submit.	Display validation error and unable to submit.	Pass	Medium	N/A
F13	Register with existing email	Navigate to Register screen. Enter existing email and submit.	Display “Email already in use” error message and unable to submit.	Pass	Medium	N/A
F14	Register with Google	Tap “Sign In with Google” button and select a Google account.	Logged in successfully using Google credentials.	Pass	High	N/A
F15	Login valid credentials	Navigate to Login screen. Enter registered email and password. Tap “Login” button.	Logged in successfully and redirected to Home screen.	Pass	High	N/A
F16	Login invalid credentials	Navigate to Login screen. Enter incorrect password. Tap “Login” button.	Display “Invalid credentials” error message and unable to log in.	Pass	High	Should not reveal if email exists
F17	Forgot password	Tap “Forgot Password” in Login screen. Enter valid registered email.	Reset link is sent to the following email.	Pass	High	Firebase sends reset email.
F18	Reset with wrong email	Tap “Forgot Password” in Login screen. Enter unregistered email.	Display “Email not found” error message and unable to reset password.	Pass	Medium	N/A

Table 4.3: Test Case Functionality - Admin CMS

Admin CMS						
ID	Test Scenario	Test Description	Expected Result	Status	Severity	Notes
F19	View user list	Log in as admin and navigate to User CMS.	All users displayed with full details.	Pass	High	N/A
F20	Add new category	Navigate to Category CMS and click “Add”. Fill form and submit.	New category saved and listed.	Pass	High	Image stored in Firebase Storage.
F21	Edit category	Navigate to Category CMS. Click on the three dots icon and select “Edit”. Edit name and save.	Category details updated in Firestore.	Pass	High	N/A
F22	Invalid object data	Add object with empty name field.	Display an error message and prevent submission.	Pass	High	N/A
F23	Delete feedback	Navigate to Feedback CMS. Click on the three dots icon and select “Delete”. Confirm the deletion.	Feedback removed in Firestore.	Pass	Medium	Should confirm before deletion.

Reliability

Table 4.4: Test Case Reliability - Waste Scanning (Google Vision API)

Waste Scanning (Google Vision API)							
ID	Test Scenario	Test Description	Test Data	Expected Result	Status	Severity	Notes
R1	Scan with same image repeatedly	Upload the same image 5 times.		Display consistent label.	Pass	High	N/A
R2	Scan image with multiple items	Upload image with multiple objects.		Display several labels.	Pass	Medium	Accuracy depends on the confidence score of the labels.
R3	Scan blurry image	Upload blurry or unclear image.		Display “No item detected” or inaccurate labels.	Pass	Medium	N/A
R4	Large image size handling	Upload image with the size of more than 30MB.	N/A	Display an error message and reject the image.	Pass	Medium	N/A
R5	Unsupported image format handling	Upload image in unsupported format.	bottle.pdf	Display error message and reject the image.	Pass	Medium	N/A
R6	Network interruption	Disconnect internet and try to upload image.	N/A	Display an error message and app does not crash.	Pass	Medium	Should retry or handle gracefully.

Table 4.5: Test Case Reliability - User Authentication

User Authentication						
ID	Test Scenario	Test Description	Expected Result	Status	Severity	Notes
R7	Network interruption	Disconnect internet and try to log in.	Display an error message and app does not crash.	Pass	Medium	Should retry or handle gracefully.

Table 4.6: Test Case Reliability - Admin CMS

Admin CMS						
ID	Test Scenario	Test Description	Expected Result	Status	Severity	Notes
R8	Network interruption	Disconnect internet and try to submit category.	Display an error message and app does not crash.	Pass	Medium	Should retry or handle gracefully.

Efficiency

Table 4.7: Test Case Efficiency - Waste Scanning (Google Vision API)

Waste Scanning (Google Vision API)						
ID	Test Scenario	Test Description	Expected Result	Status	Severity	Notes
E1	Fast scan result time	Upload image and measure response time.	Result returned within 3 seconds	Pass	Medium	Measures performance under normal conditions.

Table 4.8: Test Case Efficiency -User Authentication

User Authentication						
ID	Test Scenario	Test Description	Expected Result	Status	Severity	Notes
E2	Time to login	Complete login with valid input and measure response time.	Result returned within 3 seconds	Pass	Medium	Measures performance under normal conditions.

Table 4.9: Test Case Efficiency - Admin CMS

Admin CMS						
ID	Test Scenario	Test Description	Expected Result	Status	Severity	Notes
E3	Load list quickly	Go to Object CMS with 300+ records and measure response time.	Result returned within 5 seconds	Pass	Medium	N/A

4.3.3 Usability Testing

A user testing session was conducted using a Google Form to assess the usability of the waste sorting mobile application. A total of 32 responses were collected from users who interacted with various features of the app. The goal was to evaluate the app's ease of use, visual clarity, navigation, and overall user experience.

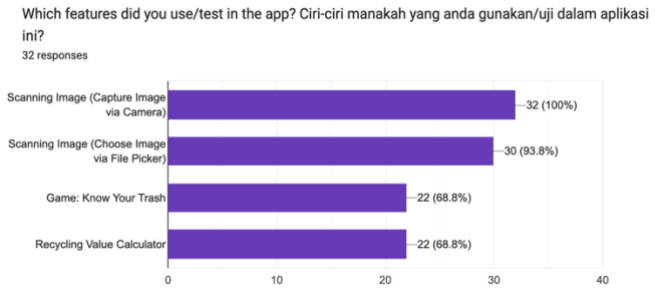


Figure 4.13: Test Results on Features Tested by Respondents

Figure 4.13 shows that all 32 respondents used the "Scanning Image (Capture Image via Camera)" feature that makes it the most tested feature. The image selection via file picker was used by 30 respondents. The interactive game "Know Your Trash" and the "Recycling Value Calculator" were both used by 22 users.

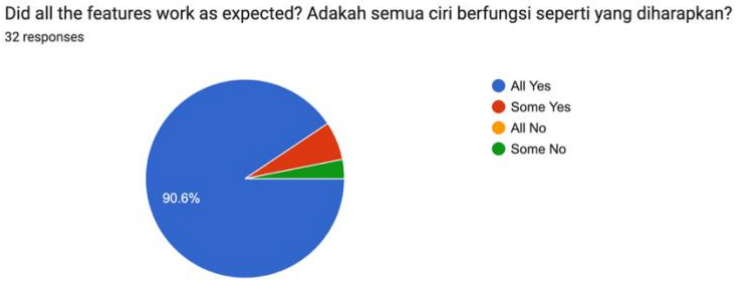


Figure 4.14: Test Results on Whether All Features Worked as Expected

Figure 4.14 shows that a large majority of respondents stated that all features worked as expected. Two users reported that only some features worked, and one user reported that some features did not work.

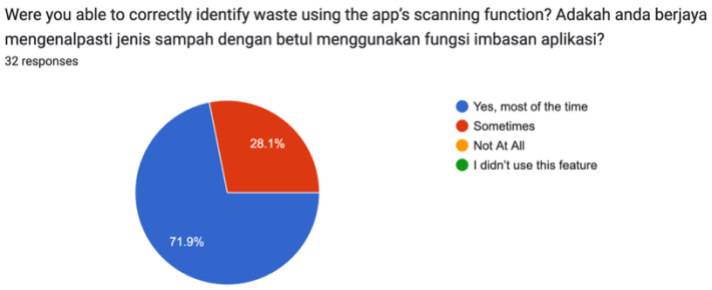


Figure 4.15: Test Results on User Ability to Identify Waste using Scanning Function

Figure 4.15 shows that most users were able to correctly identify waste using the scanning feature. Nine users selected "sometimes", indicating that while the recognition was fairly accurate, some images may have caused confusion or misidentification.

Figure 4.16: Test Results on Accuracy of Scanning Result

Figure 4.16 shows that 31 out of 32 respondents agreed that the scanning result was accurate. Only one respondent reported inaccuracy. This reflects high confidence in the system's classification performance.

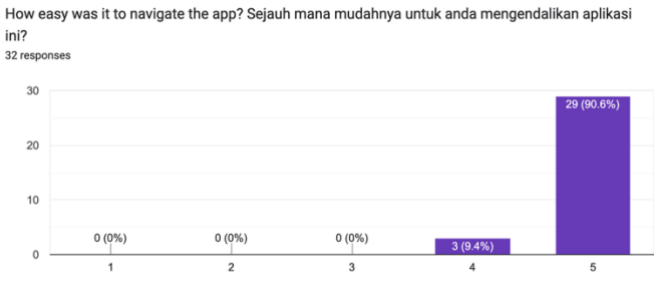


Figure 4.17: Test Results on Ease of Navigation

Figure 4.17 shows that users found the app very easy to navigate. Out of 32 responses, 29 rated the navigation as very easy. No one rated it below 4 which confirms the strong usability design.

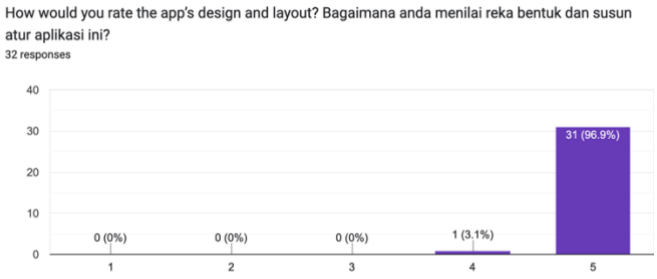


Figure 4.18: Test Results on the App's Design and Layout

Figure 4.18 shows a similarly positive view of the app's user interface. 31 users rated the design and layout as "5", and 1 rated it "4". This suggests that the visual design and layout effectively support the user experience.

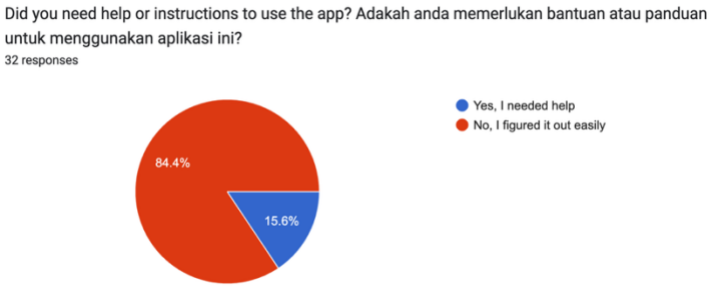


Figure 4.19: Test Results on Whether Needed Instructions to Use the App

Figure 4.19 shows that 27 users were able to use the app without help, while 5 users required assistance. While the majority found the app intuitive, this also suggests that onboarding or help screens could benefit new users.

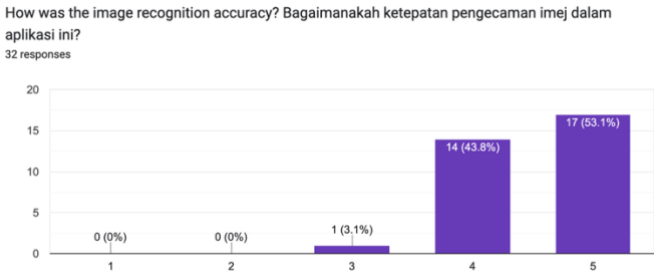


Figure 4.20: Test Results on User-rated Image Recognition Accuracy

Figure 4.20 shows that 17 users gave a "4" and 14 gave a "5", indicating that most found the image recognition quite accurate. One user rated it "3," which may reflect occasional inconsistencies in label detection.

Figure 4.21: Test Results on Willingness to Use the App if Available on Play Store

Figure 4.21 shows that 23 respondents would use the app in real life, while 9 selected "Maybe." No users rejected the idea, indicating strong user interest in the app's continued use and distribution.

Figure 4.22: Test Results on User Suggestions for Future Updates

Figure 4.22 shows the open-ended feedback from users. Some suggestions for improvement include adding a “nearby recycling location” feature and enhancing material recognition, such as distinguishing glass from plastic bottles. These responses provide direction for future feature enhancements.

4.3.4 Sample Recognition Results Using Google Vision API

Figure 4.23 shows the confidence scores from the Google Vision API when recognizing individual waste items with a white background. The results show a moderately high average accuracy, though some scores fall below 0.70, likely due to small or less visually distinct items. The uniform white background generally supports recognition but may not provide sufficient contrast for every object.

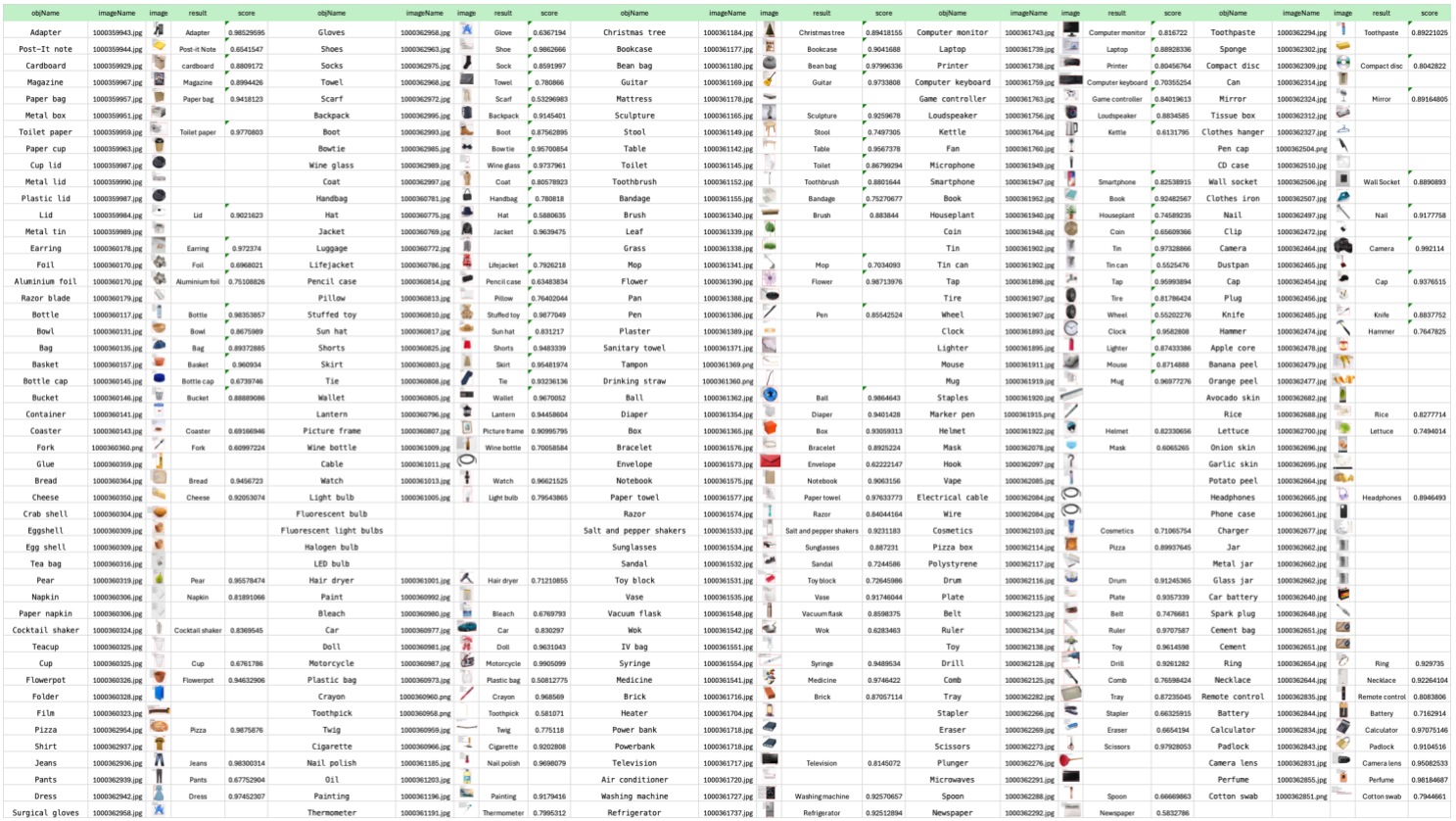
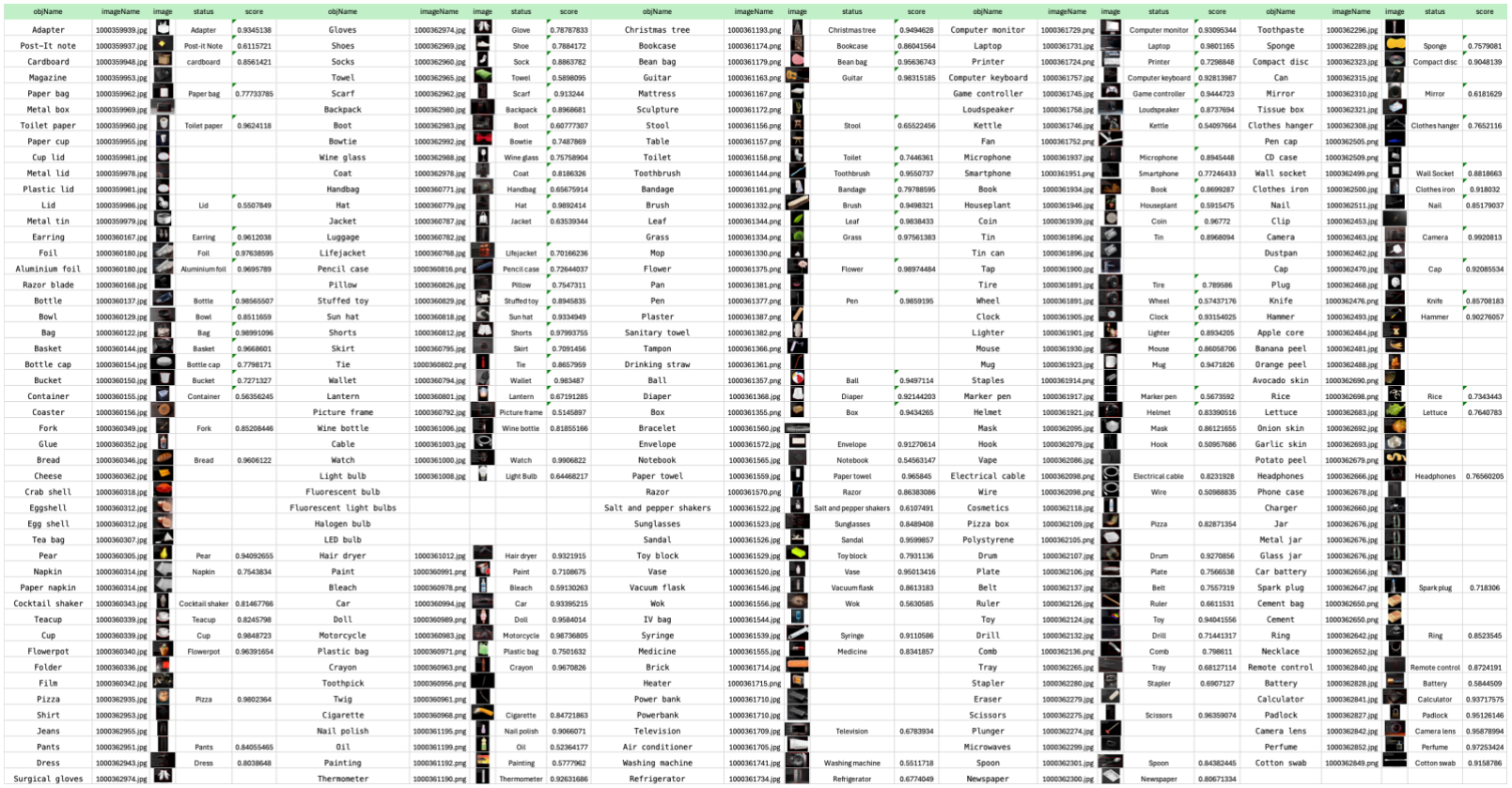


Figure 4.23: Recognition Results - Single Item with White Background

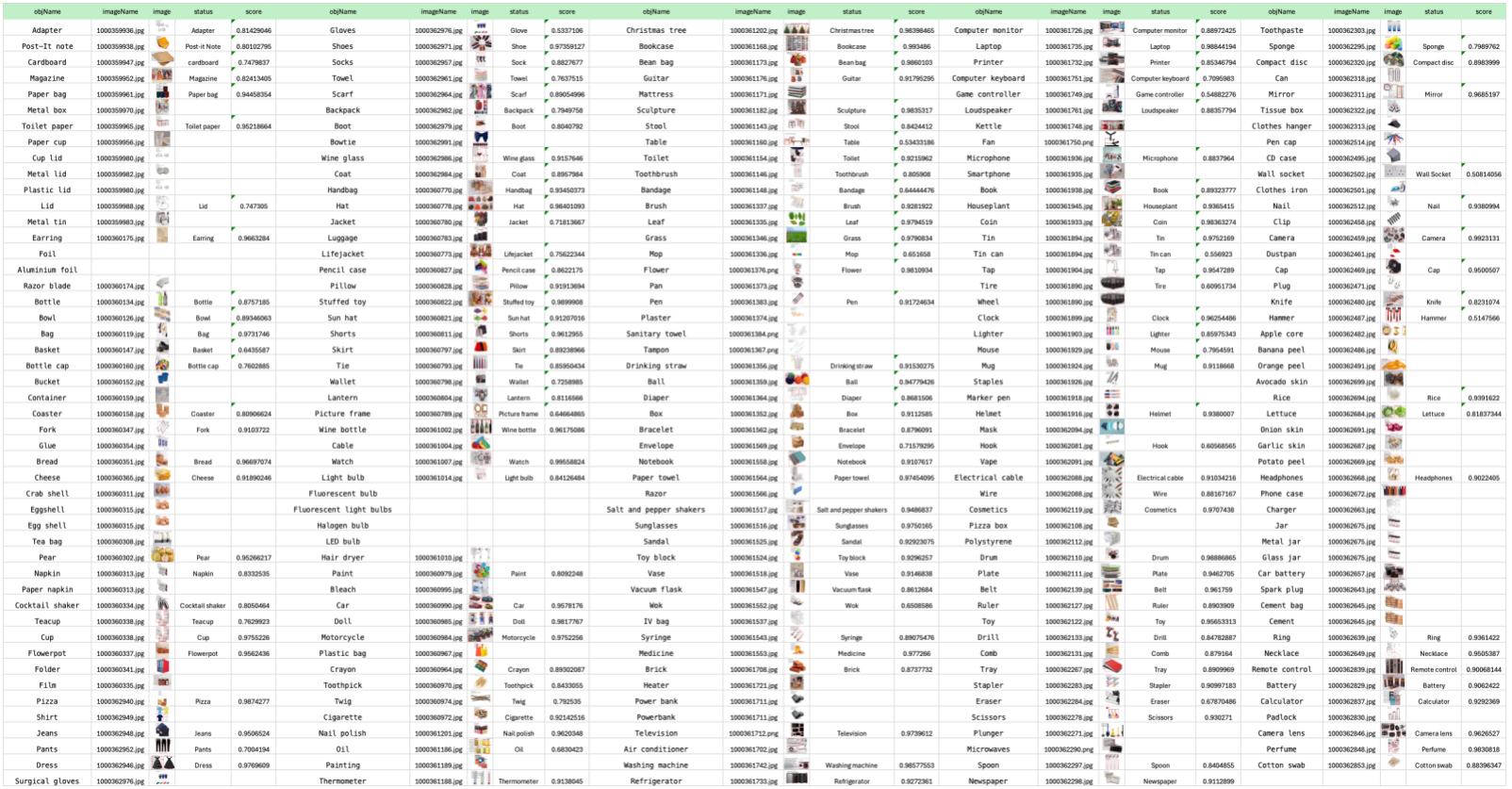
Figure 4.24 shows the results for individual items that were placed on a dark background to observe any variation in detection performance. As seen in the figure, while several confidence scores remain high, the overall average is slightly lower than with a white background. Each dark background of each image does not provide the same contrast which might make difference to the confidence scores.



objName	imageName	image	status	score	objName	imageName	image	status	score	objName	imageName	image	status	score	objName	imageName	image	status	score	objName	imageName	image	status	score
Adapter	1000359939.jpg		Adapter	0.9345138	Gloves	1000362974.jpg		Glove	0.78787833	Christmas tree	1000361193.png		Christmas tree	0.9494628	Computer monitor	1000361729.png		Computer monitor	0.93095344	Toothpaste	1000362296.jpg			
Post-It note	1000359937.jpg		Post-it Note	0.6115721	Shoes	1000362969.jpg		Shoe	0.7884172	Bookcase	1000361174.png		Bookcase	0.86041564	Laptop	1000361731.jpg		Laptop	0.9801165	Sponge	1000362289.jpg		Sponge	0.7579081
Cardboard	1000359948.jpg		cardboard	0.8561421	Socks	1000362960.jpg		Sock	0.8863782	Bean bag	1000361179.png		Bean bag	0.95636743	Printer	1000361724.png		Printer	0.7298848	Compact disc	1000362323.jpg		Compact disc	0.9048139
Magazine	1000359963.jpg				Towel	1000362965.jpg		Towel	0.5898095	Guitar	1000361163.png		Guitar	0.98315185	Computer keyboard	1000361757.jpg		Computer keyboard	0.92813987	Can	1000362315.jpg			
Paper bag	1000359962.jpg		Paper bag	0.77733785	Scarf	1000362962.jpg		Scarf	0.913244	Mattress	1000361167.png				Game controller	1000361745.jpg		Game controller	0.9444723	Mirror	1000362310.jpg		Mirror	0.6181629
Metal box	1000359969.jpg				Backpack	1000362980.jpg		Backpack	0.8968681	Sculpture	1000361172.png				Loudspeaker	1000361758.jpg		Loudspeaker	0.8737694	Tissue box	1000362321.jpg			
Toilet paper	1000359960.jpg		Toilet paper	0.9624118	Boot	1000362983.jpg		Boot	0.60777307	Stool	1000361156.png		Stool	0.65522456	Kettle	1000361746.jpg		Kettle	0.54097664	Clothes hanger	1000362308.jpg		Clothes hanger	0.7652116
Paper cup	1000359965.jpg				Bowtie	1000362992.jpg		Bowtie	0.7487869	Table	1000361157.png				Fan	1000361752.png				Pen cap	1000362505.jpg			
Cup lid	1000359981.jpg				Wine glass	1000362988.jpg		Wine glass	0.75758904	Toilet	1000361158.png		Toilet	0.7446361	Microphone	1000361937.jpg		Microphone	0.8945448	CD case	1000362509.jpg			
Metal lid	1000359978.jpg				Coat	1000362978.jpg		Coat	0.8186326	Toothbrush	1000361144.png		Toothbrush	0.9550737	Smartphone	1000361951.png		Smartphone	0.77246433	Wall socket	1000362499.png		Wall Socket	0.8818663
Plastic lid	1000359981.jpg				Handbag	1000360771.jpg		Handbag	0.65675914	Bandage	1000361161.png		Bandage	0.79788595	Book	1000361934.jpg		Book	0.8699287	Clothes iron	1000362500.jpg		Clothes iron	0.918032
Lid	1000359986.jpg		Lid	0.5507849	Hat	1000360779.jpg		Hat	0.9892414	Brush	1000361332.png		Brush	0.9498321	Houseplant	1000361946.jpg		Houseplant	0.5915475	Nail	1000362511.jpg		Nail	0.85179037
Metal tin	1000359979.jpg				Jacket	1000360787.jpg		Jacket	0.63539344	Leaf	1000361344.png		Leaf	0.9838433	Coin	1000361939.jpg		Coin	0.96772	Clip	1000362453.jpg			
Earring	1000360167.jpg		Earring	0.9612038	Luggage	1000360782.jpg				Grass	1000361334.png		Grass	0.97561383	Tin	1000361896.jpg		Tin	0.8968094	Camera	1000362463.jpg		Camera	0.9920813
Foil	1000360180.jpg		Foil	0.97638595	Lifejacket	1000360768.jpg		Lifejacket	0.70166236	Mop	1000361330.png				Tin can	1000361896.jpg				Dustpan	1000362462.jpg			
Aluminium foil	1000360180.jpg		Aluminium foil	0.9695789	Pencil case	1000360816.png		Pencil case	0.72644037	Flower	1000361375.png		Flower	0.98974484	Tap	1000361900.jpg				Cap	1000362470.jpg		Cap	0.92085534
Razor blade	1000360168.jpg				Pillow	1000360826.jpg		Pillow	0.7547311	Pan	1000361381.png				Tire	1000361891.jpg		Tire	0.789586	Plug	1000362468.jpg			
Bottle	1000360137.jpg		Bottle	0.98565507	Stuffed toy	1000360829.jpg		Stuffed toy	0.8945835	Pen	1000361377.png		Pen	0.9859195	Wheel	1000361891.jpg		Wheel	0.57437176	Knife	1000362476.jpg		Knife	0.85708183
Bowl	1000360129.jpg		Bowl	0.8511659	Sun hat	1000360818.jpg		Sun hat	0.9334949	Plaster	1000361387.png				Clock	1000361905.jpg		Clock	0.93154025	Hammer	1000362493.jpg		Hammer	0.90276057
Bag	1000360122.jpg		Bag	0.98991096	Shorts	1000360812.jpg		Shorts	0.97993755	Sanitary towel	1000361382.png				Lighter	1000361901.jpg		Lighter	0.8934205	Apple core	1000362484.jpg			
Basket	1000360144.jpg		Basket	0.9668601	Skirt	1000360795.jpg		Skirt	0.7091456	Tampon	1000361366.png				Mouse	1000361930.jpg		Mouse	0.86058706	Banana peel	1000362481.jpg			
Bottle cap	1000360154.jpg		Bottle cap	0.7798171	Tie	1000360802.png		Tie	0.8657959	Drinking straw	1000361361.png				Mug	1000361923.jpg		Mug	0.9471826	Orange peel	1000362488.jpg			
Bucket	1000360150.jpg		Bucket	0.7271327	Wallet	1000360794.jpg		Wallet	0.983487	Ball	1000361357.png		Ball	0.9497114	Staples	1000361914.png				Avocado skin	1000362690.png			
Container	1000360155.jpg		Container	0.56356245	Lantern	1000360801.jpg		Lantern	0.67191285	Diaper	1000361368.jpg		Diaper	0.92144203	Marker pen	1000361917.jpg		Marker pen	0.5673592	Rice	1000362698.png		Rice	0.7343443
Coaster	1000360156.jpg				Picture frame	1000360792.jpg		Picture frame	0.5145897	Box	1000361355.png		Box	0.9434265	Helmet	1000361921.jpg		Helmet	0.83390516	Lettuce	1000362683.jpg		Lettuce	0.7640783
Fork	1000360349.jpg		Fork	0.85208446	Wine bottle	1000361006.jpg		Wine bottle	0.81855166	Bracelet	1000361560.jpg				Mask	1000362095.jpg		Mask	0.86121655	Onion skin	1000362692.jpg			
Glue	1000360352.jpg				Cable	1000361003.jpg				Envelope	1000361572.jpg		Envelope	0.91270614	Hook	1000362079.jpg		Hook	0.50957686	Garlic skin	1000362693.jpg			
Bread	1000360346.jpg		Bread	0.9606122	Watch	1000361000.jpg		Watch	0.9906822	Notebook	1000361565.jpg		Notebook	0.54563147	Vape	1000362086.jpg				Potato peel	1000362679.png			
Cheese	1000360362.jpg				Light bulb	1000361008.jpg		Light Bulb	0.64468217	Paper towel	1000361559.jpg		Paper towel	0.965845	Electrical cable	1000362098.png		Electrical cable	0.8231928	Headphones	1000362666.jpg		Headphones	0.76560205
Crab shell	1000360318.jpg				Fluorescent bulb					Razor	1000361570.jpg		Razor	0.86383086	Wire	1000362098.png		Wire	0.50988835	Phone case	1000362678.jpg			
Eggshell	1000360312.jpg				Fluorescent light bulbs					Salt and pepper shakers	1000361522.jpg		Salt and pepper shakers	0.6107491	Cosmetics	1000362118.jpg				Charger	1000362660.jpg			
Egg shell	1000360312.jpg				Halogen bulb					Sunglasses	1000361523.jpg		Sunglasses	0.8489408	Pizza box	1000362109.jpg		Pizza	0.82871354	Jar	1000362676.jpg			
Tea bag	1000360307.jpg				LED bulb					Sandal	1000361526.jpg		Sandal	0.9599857	Polystyrene	1000362105.png				Metal jar	1000362676.jpg			
Pear	1000360305.jpg		Pear	0.94092655	Hair dryer	1000361012.jpg		Hair dryer	0.9321915	Toy block	1000361529.jpg		Toy block	0.7931136	Drum	1000362107.jpg		Drum	0.9270856	Glass jar	1000362676.jpg			
Napkin	1000360314.jpg		Napkin	0.7543834	Paint	1000360991.jpg		Paint	0.7108675	Vase	1000361620.jpg		Vase	0.95013416	Plate	1000362106.jpg		Plate	0.7566538	Car battery	1000362656.jpg			
Paper napkin	1000360314.jpg				Bleach	1000360978.png		Bleach	0.59130263	Vacuum flask	1000361546.jpg		Vacuum flask	0.8613183	Belt	1000362137.jpg		Belt	0.7557319	Spark plug	1000362647.jpg		Spark plug	0.718306
Cocktail shaker	1000360343.jpg		Cocktail shaker	0.81467766	Car	1000360994.jpg		Car	0.93395215	Wok	1000361556.jpg		Wok	0.5630585	Ruler	1000362126.jpg		Ruler	0.6611531	Cement bag	1000362650.jpg			
Teacup	1000360339.jpg		Teacup	0.8245798	Doll	1000360989.jpg		Doll	0.9584014	IV bag	1000361544.jpg				Toy	1000362124.jpg		Toy	0.94041556	Cement	1000362650.jpg			
Cup	1000360339.jpg		Cup	0.9848723	Motorcycle	1000360983.jpg		Motorcycle	0.98736805	Syringe	1000361539.jpg		Syringe	0.9110586	Drill	1000362132.jpg		Drill	0.71441317	Ring	1000362642.jpg		Ring	0.8523545
Flowerpot	1000360340.jpg		Flowerpot	0.96391654	Plastic bag	1000360971.jpg		Plastic bag	0.7501632	Medicine	1000361555.jpg		Medicine	0.8341857	Comb	1000362136.png		Comb	0.798611	Necklace	1000362652.jpg			
Folder	1000360336.jpg				Crayon	1000360963.png		Crayon	0.9670826	Brick	1000361714.jpg				Tray	1000362265.jpg		Tray	0.68127114	Remote control	1000362840.jpg		Remote control	0.8724191
Film	1000360342.jpg				Toothpick	1000360956.jpg				Heater	1000361715.png				Stapler	1000362280.jpg		Stapler	0.6907127	Battery	1000362828.jpg		Battery	0.5844509
Pizza	1000362935.jpg		Pizza	0.9802364	Twig	1000360961.png				Power bank	1000361710.jpg				Eraser	1000362279.jpg				Calculator	1000362841.jpg		Calculator	0.93717575
Shirt	1000362953.jpg				Cigarette	1000360968.png		Cigarette	0.84721863	Powerbank	1000361710.jpg				Scissors	1000362275.jpg		Scissors	0.96359074	Padlock	1000362827.jpg		Padlock	0.95126146
Jeans	1000362955.jpg				Nail polish	1000361195.png		Nail polish	0.9066071	Television	1000361709.jpg		Television	0.6783934	Plunger	1000362274.jpg				Camera lens	1000362842.jpg		Camera lens	0.95878994
Pants	1000362951.jpg		Pants	0.84055465	Oil	1000361199.png		Oil	0.52364177	Air conditioner	1000361705.jpg				Microwaves	1000362299.jpg				Perfume	1000362852.jpg		Perfume	0.97253424
Dress	1000362943.jpg		Dress	0.8038648	Painting	1000361192.png		Painting	0.5777962	Washing machine	1000361741.jpg		Washing machine	0.5511718	Spoon	1000362301.jpg		Spoon	0.84382445	Cotton swab	1000362849.jpg		Cotton swab	0.9158786
Surgical gloves	1000362974.jpg				Thermometer	1000361190.png		Thermometer	0.92631686	Refrigerator	1000361734.jpg		Refrigerator	0.6774049	Newspaper	1000362300.jpg		Newspaper	0.80671334					

Figure 4.24: Recognition Results - Single Item with Dark Background

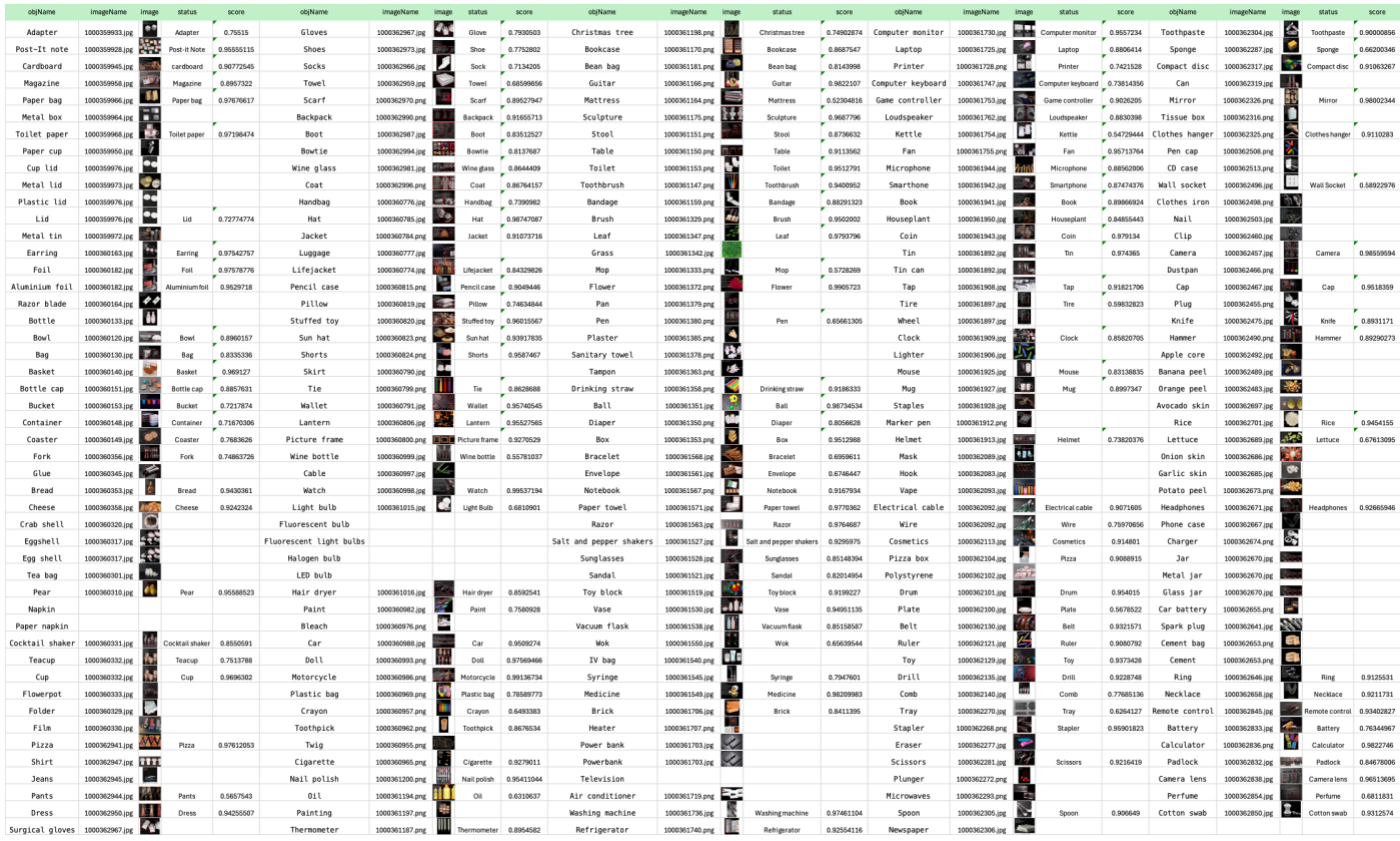
Figure 4.25 shows the results when multiple identical or similar items are shown together on a white background. The confidence scores are generally higher. This is to prove that the presence of repeated visual patterns helps the model recognize objects more effectively.



objName	imageName	image	status	score	objName	imageName	image	status	score	objName	imageName	image	status	score	objName	imageName	image	status	score	objName	imageName	image	status	score
Adapter	1000359936.jpg		Adapter	0.81429046	Gloves	1000362976.jpg		Glove	0.5337106	Christmas tree	1000361202.jpg		Christmas tree	0.98398465	Computer monitor	1000361726.jpg		Computer monitor	0.88972425	Toothpaste	1000362303.jpg			
Post-It note	1000359938.jpg		Post-it Note	0.80102795	Shoes	1000362971.jpg		Shoe	0.97359127	Bookcase	1000361168.jpg		Bookcase	0.993486	Laptop	1000361735.jpg		Laptop	0.98844194	Sponge	1000362295.jpg		Sponge	0.7989762
Cardboard	1000359947.jpg		cardboard	0.7479837	Socks	1000362957.jpg		Sock	0.8827677	Bean bag	1000361173.jpg		Bean bag	0.9860103	Printer	1000361732.jpg		Printer	0.85346794	Compact disc	1000362320.jpg		Compact disc	0.8983999
Magazine	1000359962.jpg		Magazine	0.82413405	Towel	1000362961.jpg		Towel	0.7637515	Guitar	1000361176.jpg		Guitar	0.91795295	Computer keyboard	1000361751.jpg		Computer keyboard	0.7095983	Can	1000362318.jpg			
Paper bag	1000359961.jpg		Paper bag	0.94458354	Scarf	1000362964.jpg		Scarf	0.89054996	Mattress	1000361171.jpg				Game controller	1000361749.jpg		Game controller	0.54882276	Mirror	1000362311.jpg		Mirror	0.9685197
Metal box	1000359970.jpg				Backpack	1000362982.jpg		Backpack	0.7949758	Sculpture	1000361182.jpg		Sculpture	0.9835317	Loudspeaker	1000361761.jpg		Loudspeaker	0.88357794	Tissue box	1000362322.jpg			
Toilet paper	1000359965.jpg		Toilet paper	0.95218664	Boot	1000362979.jpg		Boot	0.8040792	Stool	1000361143.jpg		Stool	0.8424412	Kettle	1000361748.jpg				Clothes hanger	1000362313.jpg			
Paper cup	1000359956.jpg				Bowtie	1000362991.jpg				Table	1000361160.jpg		Table	0.53433186	Fan	1000361750.png				Pen cap	1000362514.jpg			
Cup lid	1000359980.jpg				Wine glass	1000362986.jpg		Wine glass	0.9157646	Toilet	1000361154.jpg		Toilet	0.9215962	Microphone	1000361936.jpg		Microphone	0.8837964	CD case	1000362495.jpg			
Metal lid	1000359982.jpg				Coat	1000362984.jpg		Coat	0.8957984	Toothbrush	1000361146.jpg		Toothbrush	0.805908	Smartphone	1000361935.jpg				Wall socket	1000362502.jpg		Wall Socket	0.50814056
Plastic lid	1000359980.jpg				Handbag	1000360770.jpg		Handbag	0.93450373	Bandage	1000361148.jpg		Bandage	0.64444476	Book	1000361938.jpg		Book	0.89323777	Clothes iron	1000362501.jpg			
Lid	1000359988.jpg		Lid	0.747305	Hat	1000360778.jpg		Hat	0.98401093	Brush	1000361337.jpg		Brush	0.9281922	Houseplant	1000361945.jpg		Houseplant	0.9365415	Nail	1000362512.jpg		Nail	0.9380994
Metal tin	1000359983.jpg				Jacket	1000360780.jpg		Jacket	0.71813667	Leaf	1000361335.jpg		Leaf	0.9794519	Coin	1000361933.jpg		Coin	0.98363274	Clip	1000362458.jpg			
Earring	1000360175.jpg		Earring	0.9663284	Luggage	1000360783.jpg				Grass	1000361346.jpg		Grass	0.9790834	Tin	1000361894.jpg		Tin	0.9752169	Camera	1000362459.jpg		Camera	0.9923131
Foil					Lifejacket	1000360773.jpg		Lifejacket	0.75622344	Mop	1000361336.jpg		Mop	0.651658	Tin can	1000361894.jpg		Tin can	0.556923	Dustpan	1000362461.jpg			
Aluminium foil					Pencil case	1000360827.jpg		Pencil case	0.8622175	Flower	1000361376.png		Flower	0.9810934	Tap	1000361904.jpg		Tap	0.9547289	Cap	1000362469.jpg		Cap	0.9500507
Razor blade	1000360174.jpg				Pillow	1000360828.jpg		Pillow	0.91913694	Pan	1000361373.jpg				Tire	1000361890.jpg		Tire	0.60951734	Plug	1000362471.jpg			
Bottle	1000360134.jpg		Bottle	0.8757185	Stuffed toy	1000360822.jpg		Stuffed toy	0.9899908	Pen	1000361383.jpg		Pen	0.91724634	Wheel	1000361890.jpg				Knife	1000362480.jpg		Knife	0.8231074
Bowl	1000360126.jpg		Bowl	0.89346063	Sun hat	1000360821.jpg		Sun hat	0.91207016	Plaster	1000361374.jpg				Clock	1000361899.jpg		Clock	0.96254486	Hammer	1000362487.jpg		Hammer	0.5147566
Bag	1000360119.jpg		Bag	0.9731746	Shorts	1000360811.jpg		Shorts	0.9612955	Sanitary towel	1000361384.png				Lighter	1000361903.jpg		Lighter	0.85975343	Apple core	1000362482.jpg			
Basket	1000360147.jpg		Basket	0.6435587	Skirt	1000360797.jpg		Skirt	0.89238966	Tampon	1000361367.png				Mouse	1000361929.jpg		Mouse	0.7964591	Banana peel	1000362486.jpg			
Bottle cap	1000360160.jpg		Bottle cap	0.7602885	Tie	1000360793.jpg		Tie	0.85950434	Drinking straw	1000361356.jpg		Drinking straw	0.91530275	Mug	1000361924.jpg		Mug	0.9118668	Orange peel	1000362491.jpg			
Bucket	1000360152.jpg				Wallet	1000360798.jpg		Wallet	0.7258985	Ball	1000361359.jpg		Ball	0.94779426	Staples	1000361926.jpg				Avocado skin	1000362699.jpg			
Container	1000360159.jpg				Lantern	1000360804.jpg		Lantern	0.8116566	Diaper	1000361364.jpg		Diaper	0.8681506	Marker pen	1000361918.jpg				Rice	1000362694.jpg		Rice	0.9391622
Coaster	1000360158.jpg		Coaster	0.80906624	Picture frame	1000360789.jpg		Picture frame	0.64664865	Box	1000361352.jpg		Box	0.9112585	Helmet	1000361916.jpg		Helmet	0.9380007	Lettuce	1000362684.jpg		Lettuce	0.81837344
Fork	1000360347.jpg		Fork	0.9103722	Wine bottle	1000361002.jpg		Wine bottle	0.96175086	Bracelet	1000361562.jpg		Bracelet	0.8796091	Mask	1000362094.jpg				Onion skin	1000362691.jpg			
Glue	1000360354.jpg				Cable	1000361004.jpg				Envelope	1000361569.jpg		Envelope	0.71579295	Hook	1000362081.jpg		Hook	0.60568565	Garlic skin	1000362687.jpg			
Bread	1000360351.jpg		Bread	0.96697074	Watch	1000361007.jpg		Watch	0.96558824	Notebook	1000361558.jpg		Notebook	0.9107617	Vape	1000362091.jpg				Potato peel	1000362669.jpg			
Cheese	1000360365.jpg		Cheese	0.91890246	Light bulb	1000361014.jpg		Light bulb	0.84126484	Paper towel	1000361564.jpg		Paper towel	0.97454095	Electrical cable	1000362088.jpg		Electrical cable	0.91034216	Headphones	1000362668.jpg		Headphones	0.9022405
Crab shell	1000360311.jpg				Fluorescent bulb					Razor	1000361566.jpg				Wire	1000362088.jpg		Wire	0.88167167	Phone case	1000362672.jpg			
Eggshell	1000360315.jpg				Fluorescent light bulbs					Salt and pepper shakers	1000361517.jpg		Salt and pepper shakers	0.9486837	Cosmetics	1000362119.jpg		Cosmetics	0.9707438	Charger	1000362663.jpg			
Egg shell	1000360315.jpg				Halogen bulb					Sunglasses	1000361516.jpg		Sunglasses	0.9750165	Pizza box	1000362108.jpg				Jar	1000362675.jpg			
Tea bag	1000360308.jpg				LED bulb					Sandal	1000361525.jpg		Sandal	0.92923075	Polystyrene	1000362112.jpg				Metal jar	1000362675.jpg			
Pear	1000360302.jpg		Pear	0.95266217	Hair dryer	1000361010.jpg				Toy block	1000361524.jpg		Toy block	0.9296257	Drum	1000362110.jpg		Drum	0.98886865	Glass jar	1000362675.jpg			
Napkin	1000360313.jpg		Napkin	0.8332535	Paint	1000360979.jpg		Paint	0.8092248	Vase	1000361518.jpg		Vase	0.9146838	Plate	1000362111.jpg		Plate	0.9462705	Car battery	1000362657.jpg			
Paper napkin	1000360313.jpg				Bleach	1000360995.jpg				Vacuum flask	1000361547.jpg		Vacuum flask	0.8612684	Belt	1000362139.jpg		Belt	0.961759	Spark plug	1000362643.jpg			
Cocktail shaker	1000360334.jpg		Cocktail shaker	0.8050464	Car	1000360990.jpg		Car	0.9578176	Wok	1000361552.jpg		Wok	0.6508586	Ruler	1000362127.jpg		Ruler	0.8903909	Cement bag	1000362645.jpg			
Teacup	1000360338.jpg		Teacup	0.7629923	Doll	1000360985.jpg		Doll	0.9817767	IV bag	1000361537.jpg				Toy	1000362122.jpg		Toy	0.95653313	Cement	1000362645.jpg			
Cup	1000360338.jpg		Cup	0.9755226	Motorcycle	1000360984.jpg		Motorcycle	0.9752256	Syringe	1000361543.jpg		Syringe	0.89075476	Drill	1000362133.jpg		Drill	0.84782887	Ring	1000362639.jpg		Ring	0.9361422
Flowerpot	1000360337.jpg		Flowerpot	0.9662436	Plastic bag	1000360967.jpg				Medicine	1000361553.jpg		Medicine	0.977266	Comb	1000362131.jpg		Comb	0.879164	Necklace	1000362649.jpg		Necklace	0.9505387
Folder	1000360341.jpg				Crayon	1000360964.jpg		Crayon	0.89302087	Brick	1000361708.jpg		Brick	0.8737732	Tray	1000362267.jpg		Tray	0.8909969	Remote control	1000362839.jpg		Remote control	0.90068144
Film	1000360335.jpg				Toothpick	1000360970.jpg		Toothpick	0.8433055	Heater	1000361721.jpg				Stapler	1000362283.jpg		Stapler	0.90997183	Battery	1000362829.jpg		Battery	0.9062422
Pizza	1000362940.jpg		Pizza	0.9874277	Twig	1000360974.jpg		Twig	0.792535	Power bank	1000361711.jpg				Eraser	1000362284.jpg		Eraser	0.67870486	Calculator	1000362837.jpg		Calculator	0.9292369
Shirt	1000362949.jpg				Cigarette	1000360972.jpg		Cigarette	0.92142516	Powerbank	1000361711.jpg				Scissors	1000362278.jpg		Scissors	0.930271	Padlock	1000362830.jpg			
Jeans	1000362948.jpg		Jeans	0.9506524	Nail polish	1000361201.jpg		Nail polish	0.9620348	Television	1000361712.png		Television	0.9739612	Plunger	1000362271.jpg				Camera lens	1000362846.jpg		Camera lens	0.9626527
Pants	1000362952.jpg		Pants	0.7004194	Oil	1000361186.jpg		Oil	0.6830423	Air conditioner	1000361702.jpg				Microwaves	1000362290.png				Perfume	1000362848.jpg		Perfume	0.9830818
Dress	1000362946.jpg		Dress	0.9769609	Painting	1000361189.jpg				Washing machine	1000361742.jpg		Washing machine	0.98577553	Spoon	1000362297.jpg		Spoon	0.8404855	Cotton swab	1000362853.jpg		Cotton swab	0.88396347
Surgical gloves	1000362976.jpg				Thermometer	1000361188.jpg		Thermometer	0.9138045	Refrigerator	1000361733.jpg		Refrigerator	0.9272361	Newspaper	1000362298.jpg		Newspaper	0.9112899					

Figure 4.25: Recognition Results - Multiple Items with White Background

Figure 4.26 shows the results when multiple identical or similar items are shown together on a dark background. Recognition accuracy remains high in this scenario with many confidence scores above 0.90, possibly because the repeated items create stronger visual pattern. However, slight drops in score can occur if object edges blend into the background.



objName	imageName	image	status	score	objName	imageName	image	status	score	objName	imageName	image	status	score	objName	imageName	image	status	score	objName	imageName	image	status	score
Adapter	1000359933.jpg		Adapter	0.75515	Gloves	1000362967.jpg		Glove	0.7930503	Christmas tree	1000361198.png		Christmas tree	0.74902874	Computer monitor	1000361730.jpg		Computer monitor	0.9557234	Toothpaste	1000362304.jpg		Toothpaste	0.90000856
Post-It note	1000359928.jpg		Post-it Note	0.95555115	Shoes	1000362973.jpg		Shoe	0.7752802	Bookcase	1000361170.png		Bookcase	0.8687547	Laptop	1000361725.jpg		Laptop	0.8806414	Sponge	1000362287.jpg		Sponge	0.66200346
Cardboard	1000359945.jpg		cardboard	0.90772545	Socks	1000362966.jpg		Sock	0.7134205	Bean bag	1000361181.png		Bean bag	0.8143998	Printer	1000361728.jpg		Printer	0.7421528	Compact disc	1000362317.jpg		Compact disc	0.91063267
Magazine	1000359958.jpg		Magazine	0.8957322	Towel	1000362959.jpg		Towel	0.68599856	Guitar	1000361166.png		Guitar	0.9822107	Computer keyboard	1000361747.jpg		Computer keyboard	0.73814356	Can	1000362319.jpg			
Paper bag	1000359966.jpg		Paper bag	0.97676617	Scarf	1000362970.png		Scarf	0.89527947	Mattress	1000361164.png		Mattress	0.52304816	Game controller	1000361753.jpg		Game controller	0.9026205	Mirror	1000362326.jpg		Mirror	0.98002344
Metal box	1000359964.jpg				Backpack	1000362990.png		Backpack	0.91655713	Sculpture	1000361176.png		Sculpture	0.9687796	Loudspeaker	1000361762.jpg		Loudspeaker	0.8830398	Tissue box	1000362316.jpg			
Toilet paper	1000359968.jpg		Toilet paper	0.97198474	Boot	1000362987.jpg		Boot	0.83512527	Stool	1000361151.png		Stool	0.8736632	Kettle	1000361754.jpg		Kettle	0.54729444	Clothes hanger	1000362325.jpg		Clothes hanger	0.9110283
Paper cup	1000359950.jpg				Bowtie	1000362994.jpg		Bowtie	0.8137687	Table	1000361150.png		Table	0.9113562	Fan	1000361755.png		Fan	0.95713764	Pen cap	1000362508.jpg			
Cup lid	1000359976.jpg				Wine glass	1000362981.jpg		Wine glass	0.8644409	Toilet	1000361153.png		Toilet	0.9512791	Microphone	1000361944.jpg		Microphone	0.88562006	CD case	1000362513.jpg			
Metal lid	1000359973.jpg				Coat	1000362996.png		Coat	0.86764157	Toothbrush	1000361147.png		Toothbrush	0.9400952	Smarthone	1000361942.jpg		Smartphone	0.87474376	Wall socket	1000362496.jpg		Wall Socket	0.58922976
Plastic lid	1000359975.jpg				Handbag	1000360776.jpg		Handbag	0.7390982	Bandage	1000361159.png		Bandage	0.88291323	Book	1000361941.jpg		Book	0.89866924	Clothes iron	1000362498.jpg			
Lid	1000359976.jpg		Lid	0.72774774	Hat	1000360785.jpg		Hat	0.98747087	Brush	1000361329.png		Brush	0.9502002	Houseplant	1000361950.jpg		Houseplant	0.84855443	Nail	1000362503.jpg			
Metal tin	1000359972.jpg				Jacket	1000360784.png		Jacket	0.91073716	Leaf	1000361347.png		Leaf	0.9793796	Coin	1000361943.jpg		Coin	0.979134	Clip	1000362460.jpg			
Earring	1000360163.jpg		Earring	0.97542757	Luggage	1000360777.jpg				Grass	1000361342.jpg				Tin	1000361892.jpg		Tin	0.974365	Camera	1000362457.jpg		Camera	0.98599594
Foil	1000360182.jpg		Foil	0.97578776	Lifejacket	1000360774.jpg		Lifejacket	0.84329826	Mop	1000361333.png		Mop	0.5728269	Tin can	1000361892.jpg				Dustpan	1000362466.jpg			
Aluminium foil	1000360182.jpg		Aluminium foil	0.9529718	Pencil case	1000360815.png		Pencil case	0.9049646	Flower	1000361372.png		Flower	0.9905723	Tap	1000361908.jpg		Tap	0.91821706	Cap	1000362467.jpg		Cap	0.9518359
Razor blade	1000360164.jpg				Pillow	1000360819.jpg		Pillow	0.74634844	Pan	1000361379.png				Tire	1000361897.jpg		Tire	0.59832823	Plug	1000362455.jpg			
Bottle	1000360133.jpg				Stuffed toy	1000360820.jpg		Stuffed toy	0.96015567	Pen	1000361380.png		Pen	0.65661305	Wheel	1000361897.jpg				Knife	1000362475.jpg		Knife	0.8931171
Bowl	1000360120.jpg		Bowl	0.8960157	Sun hat	1000360823.jpg		Sun hat	0.93917835	Plaster	1000361385.png				Clock	1000361909.jpg		Clock	0.85820705	Hammer	1000362490.png		Hammer	0.89290273
Bag	1000360130.jpg		Bag	0.8335336	Shorts	1000360824.png		Shorts	0.9587467	Sanitary towel	1000361378.png				Lighter	1000361906.jpg				Apple core	1000362492.jpg			
Basket	1000360140.jpg		Basket	0.969127	Skirt	1000360790.jpg				Tampon	1000361363.png				Mouse	1000361925.jpg		Mouse	0.83138835	Banana peel	1000362489.jpg			
Bottle cap	1000360151.jpg		Bottle cap	0.8857631	Tie	1000360799.png		Tie	0.8628688	Drinking straw	1000361358.png		Drinking straw	0.9186333	Mug	1000361927.jpg		Mug	0.8997347	Orange peel	1000362483.jpg			
Bucket	1000360153.jpg		Bucket	0.7217874	Wallet	1000360791.jpg		Wallet	0.95740545	Ball	1000361351.jpg		Ball	0.98734534	Staples	1000361928.jpg				Avocado skin	1000362697.jpg			
Container	1000360148.jpg		Container	0.71670306	Lantern	1000360806.jpg		Lantern	0.95527565	Diaper	1000361350.png		Diaper	0.8056628	Marker pen	1000361912.jpg				Rice	1000362701.jpg		Rice	0.9454155
Coaster	1000360149.jpg		Coaster	0.7683626	Picture frame	1000360800.png		Picture frame	0.9270529	Box	1000361353.png		Box	0.9512988	Helmet	1000361913.jpg		Helmet	0.73820376	Lettuce	1000362689.jpg		Lettuce	0.67613095
Fork	1000360356.jpg		Fork	0.74863726	Wine bottle	1000360999.jpg		Wine bottle	0.55781037	Bracelet	1000361568.jpg		Bracelet	0.6959611	Mask	1000362089.jpg				Onion skin	1000362686.jpg			
Glue	1000360345.jpg				Cable	1000360997.jpg				Envelope	1000361561.jpg		Envelope	0.6746447	Hook	1000362083.jpg				Garlic skin	1000362685.jpg			
Bread	1000360353.jpg		Bread	0.9430361	Watch	1000360998.jpg		Watch	0.99537194	Notebook	1000361567.png		Notebook	0.9167934	Vape	1000362093.jpg				Potato peel	1000362673.jpg			
Cheese	1000360358.jpg		Cheese	0.9242324	Light bulb	1000361015.jpg		Light Bulb	0.6810901	Paper towel	1000361571.jpg		Paper towel	0.9770362	Electrical cable	1000362092.jpg		Electrical cable	0.9071605	Headphones	1000362671.jpg		Headphones	0.92665946
Crab shell	1000360320.jpg				Fluorescent bulb					Razor	1000361563.jpg		Razor	0.9764687	Wire	1000362092.jpg		Wire	0.79970656	Phone case	1000362667.jpg			
Eggshell	1000360317.jpg				Fluorescent light bulbs					Salt and pepper shakers	1000361527.jpg		Salt and pepper shakers	0.9295975	Cosmetics	1000362113.jpg		Cosmetics	0.914801	Charger	1000362674.jpg			
Egg shell	1000360317.jpg				Halogen bulb					Sunglasses	1000361528.jpg		Sunglasses	0.85148394	Pizza box	1000362104.jpg		Pizza	0.9088915	Jar	1000362670.jpg			
Tea bag	1000360301.jpg				LED bulb					Sandal	1000361521.jpg		Sandal	0.82014954	Polystyrene	1000362102.jpg				Metal jar	1000362670.jpg			
Pear	1000360310.jpg		Pear	0.95588523	Hair dryer	1000361016.jpg		Hair dryer	0.8592541	Toy block	1000361519.jpg		Toy block	0.9199227	Drum	1000362101.jpg		Drum	0.954015	Glass jar	1000362670.jpg			
Napkin					Paint	1000360982.jpg		Paint	0.7580928	Vase	1000361530.jpg		Vase	0.94951135	Plate	1000362100.jpg		Plate	0.5678522	Car battery	1000362655.jpg			
Paper napkin					Bleach	1000360976.png				Vacuum flask	1000361538.jpg		Vacuum flask	0.85158587	Belt	1000362130.jpg		Belt	0.9321571	Spark plug	1000362641.jpg			
Cocktail shaker	1000360331.jpg		Cocktail shaker	0.8550591	Car	1000360988.jpg		Car	0.9509274	Wok	1000361550.jpg		Wok	0.65639544	Ruler	1000362121.jpg		Ruler	0.9080792	Cement bag	1000362653.jpg			
Teacup	1000360332.jpg		Teacup	0.7513788	Doll	1000360993.jpg		Doll	0.97569466	IV bag	1000361540.png				Toy	1000362129.jpg		Toy	0.9373428	Cement	1000362653.jpg			
Cup	1000360332.jpg		Cup	0.9696302	Motorcycle	1000360986.png		Motorcycle	0.99136734	Syringe	1000361545.jpg		Syringe	0.7947601	Drill	1000362135.jpg		Drill	0.9228748	Ring	1000362646.jpg		Ring	0.9125531
Flowerpot	1000360333.jpg				Plastic bag	1000360969.png		Plastic bag	0.78589773	Medicine	1000361549.jpg		Medicine	0.98209983	Comb	1000362140.jpg		Comb	0.77685136	Necklace	1000362658.jpg		Necklace	0.9211731
Folder	1000360329.jpg				Crayon	1000360957.png		Crayon	0.6493383	Brick	1000361706.jpg		Brick	0.8411395	Tray	1000362270.jpg		Tray	0.6264127	Remote control	1000362845.jpg		Remote control	0.93402827
Film	1000360330.jpg				Toothpick	1000360962.png		Toothpick	0.8676534	Heater	1000361707.png				Stapler	1000362268.jpg		Stapler	0.95901823	Battery	1000362833.jpg		Battery	0.76344967
Pizza	1000362941.jpg		Pizza	0.97612053	Twig	1000360955.png				Power bank	1000361703.jpg				Eraser	1000362277.jpg				Calculator	1000362836.jpg		Calculator	0.9822746
Shirt	1000362947.jpg				Cigarette	1000360965.png		Cigarette	0.9279011	Powerbank	1000361703.jpg				Scissors	1000362281.jpg		Scissors	0.9216419	Padlock	1000362832.jpg		Padlock	0.84678006
Jeans	1000362945.jpg				Nail polish	1000361200.png		Nail polish	0.95411044	Television					Plunger	1000362272.jpg				Camera lens	1000362838.jpg		Camera lens	0.96513695
Pants	1000362944.jpg		Pants	0.5657543	Oil	1000361194.png		Oil	0.6310637	Air conditioner	1000361719.png				Microwaves	1000362293.jpg				Perfume	1000362854.jpg		Perfume	0.6811831
Dress	1000362950.jpg		Dress	0.94255507	Painting	1000361197.png				Washing machine	1000361736.jpg		Washing machine	0.97461104	Spoon	1000362305.jpg		Spoon	0.906649	Cotton swab	1000362850.jpg		Cotton swab	0.9312574
Surgical gloves	1000362967.jpg				Thermometer	1000361187.png		Thermometer	0.8954582	Refrigerator	1000361740.png		Refrigerator	0.92554116	Newspaper	1000362306.jpg								

Figure 4.26: Recognition Results - Multiple Items with Dark Background

Table 4.10: Confidence Score Summary Across Image Scenarios

Test Scenario	Total Samples	Min Confidence	Max Confidence	Avg. Confidence	Dominant Score Range
Single Item with White Background	138	0.508	0.992	~ 0.845	0.80 - 0.98
Single Item with Dark Background	138	0.509	0.992	~ 0.835	0.75 - 0.98
Multiple Items with White Background	135	0.508	0.995	~ 0.869	0.80 - 0.98
Multiple Items with Dark Background	139	0.523	0.995	~ 0.867	0.75 - 0.98

Table 4.10 presents a comparative analysis of the confidence scores returned by the Google Vision API for different waste image scenarios. The key metrics compared are the number of samples tested, the minimum and maximum confidence scores, the average confidence score, and the dominant confidence score range. These scenarios were chosen to assess how background colour and item quantity affect object recognition accuracy and reliability. The four scenarios include:

1) Single Item with White Background
2) Single Item with Dark Background
3) Multiple Items with White Background
4) Multiple Items with Dark Background

The single item with a white background scenario showed a relatively high average confidence score of approximately 0.845, with most scores falling in the 0.80 to 0.98 range. The white background likely helped the Google Vision API isolate the object with minimal noise or distractions, contributing to higher accuracy. This aligns with typical machine vision performance, where clean and uncluttered backgrounds result in better recognition. Meanwhile, for the one with dark background, the average confidence dropped slightly to around 0.835. While the system still performed well, the darker background may have introduced minor shadows or contrast issues that affected detection precision. This demonstrates that while Google Vision can handle varying lighting conditions, white backgrounds are slightly more optimal for recognition accuracy.

For the multiple items scenario, the one with white background yielded the highest average confidence score of approximately 0.869. Interestingly, even with more than one object in the image, the Vision API still performed very well likely because the white background allowed for better distinction between individual items. The system could correctly detect the primary item or return multiple high-confidence labels. Despite the complexity of multiple objects and a dark background, this scenario still produced a high average confidence score of approximately 0.867, very close to the white background scenario. This suggests the model is fairly robust, but performance may vary depending on object colour, contrast, and lighting. The dominant confidence range of 0.75 to 0.98 remains consistent with the other tests, showing reliable classification even under challenging conditions.

However, despite this, the average score for the single item scenarios was slightly lower than in the multiple items scenarios. One possible reason for this could be the inherent ambiguity of certain small or generic items when photographed alone. For example, a single

toothpick may not provide enough visual detail or context for accurate identification, while a group of toothpicks, as seen in the multiple items images, becomes more recognizable to the model due to repetitive visual patterns and clearer object context. Yet, accuracy may decrease when different types of objects are present in one image. For example, a mixed pile of unrelated waste can introduces visual complexity, making it harder for the model to determine a dominant object, which lowers the confidence and consistency of predictions.

Table 4.11: Toothpick Recognition Results

Scenario	Image	Confidence Score (%)
Single Item with White Background		58.1
Single Item with Dark Background		Not Detected
Multiple Items with White Background		84.3
Multiple Items with Dark Background		86.8
Mixed Pile of Unrelated Waste includes toothpicks, plastic, box		65.6

Table 4.11 shows a test that was conducted using the object toothpick across various image scenarios to observe how Google Vision API performs under different visual conditions. The toothpick test results clearly demonstrate the limitation mentioned before. A single toothpick on a white background yielded a low confidence score of 58.1%, and it was not detected at all on a dark background. In contrast, when multiple toothpicks were present, the confidence improved significantly with 84.3% on a white background and 86.8% on a dark background. The repetition and quantity helped the model recognize the object more effectively by providing more pattern reinforcement. On the other hand, accuracy tends to decrease when an image contains a mixed pile of unrelated objects. For example, the same test involving a mixed waste pile including toothpicks resulted in a lower confidence score of 65.6%. This supports the idea that visual complexity and competing features from different items reduce the model's ability to determine a dominant object. The model may struggle to prioritize the toothpick or may mislabel based on dominant features from other objects in the pile.

As shown in Figure 4.27, images containing mixed waste items often result in several undetected or misclassified items. This is due to the increased visual complexity and lack of a dominant object. When multiple unrelated items are present in a single image, the Google Vision API struggles to determine a clear classification. This contrasts with earlier scenarios like single item or multiple items of the same type, where the visual pattern were more consistent and distinct. These findings strengthen the need for clearer image input, or possibly introducing a custom-trained classifier in future work to handle complex real-world inputs more accurately.

Figure 4.27 shows the confidence cores for images containing a mix of unrelated waste items. The results show more inconsistencies and lower average scores that indicates difficulty in detecting a dominant object. Visual clutter and overlapping materials increase classification challenges and often resulting in partial or failed recognition.

Figure 4.27: Recognition Results - Mixed Waste Items

Objects	Categories	Objects	Categories	Objects	Categories	Objects	Categories
Adapter	Electronic waste	Cocktail shaker	Plastic, Metal	Lantern	Glass, General waste, Electronic waste, Metal	Power bank	Electronic waste
Aluminium foil	Dry Recyclables, General waste	Coffee cup	General waste	Laundry detergent	Plastic	Razor	Dry Recyclables, Electronic waste
Apple	Food waste, Compostable	Coffee filter	Food waste, Compostable	Leaf	General waste	Razor blade	Dry Recyclables
Avocado	Food waste, Compostable	Coffee grounds	Food waste, Compostable	LED bulb	Hazardous waste	Remote control	Electronic waste
Backpack	Textiles and clothes, General waste	Comb	Plastic, General waste	Leather	Textiles and clothes, General waste	Salt and pepper shakers	General waste
Bag	Plastic, Textiles and clothes, Paper	Compact disc	Electronic waste	Lifejacket	Textiles and clothes, General waste	Sandal	General waste
Baked goods	Food waste, Compostable	Computer keyboard	Electronic waste	Light bulb	Hazardous waste, General waste, Electronic waste	Sanitary towel	General waste
Ball	General waste	Computer monitor	Electronic waste	Lighting	Hazardous waste, Bulky waste, Electronic waste	Saucer	Glass, General waste
Bandage	General waste, Medical waste	Container	Plastic, Glass, General waste, Metal	Magazine	Dry Recyclables, Paper	Scarf	Textiles and clothes, General waste
Banana	Food waste, Compostable	Cotton	Textiles and clothes, General waste, Compostable	Marker pen	General waste	Sculpture	Bulky waste, General waste, Metal
Basket	Plastic, General waste	Cotton swab	General waste	Mask	Textiles and clothes, General waste, Medical waste	Shoes	Textiles and clothes, General waste
Battery cell	Hazardous waste, Electronic waste	Crab	Food waste, Compostable	Mason Jar	Dry Recyclables, Glass, General waste	Shirt	Textiles and clothes, General waste
Bean bag	Bulky waste	Crayon	General waste	Mattress	Bulky waste	Shorts	Textiles and clothes, General waste
Bed	Bulky waste	Cup	Plastic, General waste, Paper	Medicine	Medical waste	Shrimp	Food waste, Compostable
Beer bottle	Glass	Cup lid	Dry Recyclables, Plastic	Metal box	Dry Recyclables	Silk	Textiles and clothes, General waste
Bleach	Hazardous waste, Chemical waste	Curtain	Textiles and clothes, General waste	Metal lid	Dry Recyclables	Skirt	Textiles and clothes, General waste
Bookcase	Bulky waste	Diaper	General waste	Metal tin	Dry Recyclables	Smartphone	Electronic waste
Bones	General waste, Compostable	Doll	Hazardous waste, General waste	Microwave oven	Electronic waste	Sock	Textiles and clothes, General waste
Boot	Textiles and clothes, General waste	Dress	Textiles and clothes, General waste	Mop	General waste	Spray	Dry Recyclables, Hazardous waste, Metal
Bottle	Plastic, Glass, General waste	Drinking straw	General waste	Motorcycle	Hazardous waste, Vehicle	Staple	Dry Recyclables, Metal
Bottle cap	Plastic, Metal	Drum	Plastic, Hazardous waste, Bulky waste, Metal	Nail polish	Hazardous waste	Stuffed toy	Textiles and clothes, General waste
Bowtie	Textiles and clothes, General waste	Egg	Dry Recyclables, Plastic, Food waste, General waste, Paper, Compostable	Napkin	General waste	Suitcase	Textiles and clothes, General waste
Bowl	Plastic, Glass, General waste	Envelope	General waste, Paper	Notebook	General waste, Paper	Sun hat	Textiles and clothes, General waste
Box	General waste, Paper, Metal	Feminine hygiene	General waste	Oil	Hazardous waste	Sunglasses	General waste
Bracelet	General waste, Jewelry	Figurine	General waste	Outerwear	Textiles and clothes, General waste	Surgical gloves	Medical waste
Bread	Food waste, Compostable	File folder	Plastic, General waste, Paper	Paint	Hazardous waste	Table	Bulky waste
Brochure	Dry Recyclables, General waste, Paper	Fluorescent bulb	Hazardous waste	Painting	Hazardous waste, General waste	Table top	Bulky waste
Brush	General waste	Flower	General waste, Compostable	Pan	General waste, Metal	Tampon	General waste
Bucket	Plastic, General waste, Metal	Flowerpot	Plastic, General waste	Pants	Textiles and clothes, General waste	Tea bag	Food waste, Compostable
Camera	Electronic waste	Flyer	Dry Recyclables, Paper	Paper bag	Dry Recyclables	Thermometer	Hazardous waste, Electronic waste
Canned packaged goods	Food waste, Compostable	Food	Food waste, Compostable	Paper cup	Dry Recyclables	Tie	Textiles and clothes, General waste
Car	Hazardous waste, Vehicle	Footwear	Textiles and clothes, General waste	Paper towel	General waste	Toilet	Bulky waste
Cardboard	Dry Recyclables, General waste, Paper	Fork	Plastic, General waste, Metal	Peel	Food waste, Compostable	Toilet paper	Dry Recyclables, General waste, Paper, Compostable
Carpet	Bulky waste	Fruit	Food waste, Compostable	Pen	General waste	Toothbrush	General waste, Electronic waste, Compostable
Ceiling fan	Electronic waste	Frying pan	General waste, Metal	Pen cap	Plastic	Toothpick	General waste
Cereal box	Dry Recyclables	Furniture	Bulky waste	Pencil case	Textiles and clothes, General waste	Top	Textiles and clothes, General waste
Chandelier	Glass, Bulky waste, Electronic waste, Metal	Game controller	Electronic waste	Pear	Food waste, Compostable	Toy block	General waste
Cheese	Food waste, Compostable	Glass	Glass, General waste	Photographic film	Plastic, General waste	Towel	Textiles and clothes, General waste
Chest of drawers	Bulky waste	Glasses	General waste	Photo	Dry Recyclables	Twig	General waste
Chewing gum	General waste	Glove	Textiles and clothes, General waste	Picture frame	Glass, Bulky waste, General waste	Used bandages	Medical waste
Chime	Glass, General waste, Metal	Glue	Plastic, Hazardous waste	Pillow	Textiles and clothes, General waste	Vacuum cleaner bag	General waste
Christmas tree	Bulky waste, General waste	Grass clippings	Compostable	Pizza	Food waste, General waste, Paper, Compostable	Vacuum flask	General waste, Metal
Cigarette	General waste	Guitar	Bulky waste, Electronic waste	Plastic bag	General waste	Vase	General waste
Cleaning waste	General waste	Hair dryer	Hazardous waste, Electronic waste	Plastic bottle	Plastic	Vegetable	Food waste, Compostable
Clothes hanger	Dry Recyclables, Metal	Halogen bulb	Hazardous waste	Plate	Glass, General waste	Wallet	Textiles and clothes, General waste
Clothes iron	Electronic waste	Handbag	Textiles and clothes, General waste	Plaster	General waste	Wine bottle	Glass, General waste, Compostable, Metal
Clothing	Textiles and clothes, General waste	Hat	Textiles and clothes, General waste	Poster	General waste, Paper	Wine glass	Glass, General waste
Coat	Textiles and clothes, General waste	Jacket	Textiles and clothes, General waste	Post-It note	Dry Recyclables, Paper	Wok	Metal, General waste
Coaster	Plastic, Glass, General waste, Compostable	Jeans	Textiles and clothes, General waste	Potato	Food waste, Compostable	Wood chips	Compostable

Figure 4.28: List of Waste Objects and Their Categories

Figure 4.28 shows the reference database used to assist the waste sorting feature in categorizing various items accurately. It contains a wide array of objects commonly found in households and categorizes them based on appropriate waste disposal guidelines such as recyclable,

compostable, hazardous, electronic, bulky, or general waste. It serves as the foundation for determining the waste category from the object labels detected by the Google Vision API and also provide accurate, responsible, and automated waste disposal recommendations for users.

By scanning a large volume of images, the app is able to expand its understanding of real-world waste items, particularly those that may be obscured, mislabelled, or unique to certain environments. This also addresses a key limitation of object detection models, which may not always detect specific or lesser-known objects, especially when the item lacks clear visual features or is photographed in complex conditions. The object list within the reference database is continuously growing. As more images are scanned, new labels are encountered where some of which may not be initially recognized or classified by the API. In such cases, the system records these unidentified labels and allows them to be manually reviewed and added to the database over time. This iterative learning approach ensures the app becomes smarter and more comprehensive with continued usage.

4.4 Chapter Summary

This chapter outlined the implementation phase of the mobile application, including selected technologies, developed system modules, and user workflows. The integration of Firebase and Google Vision API allows the application to provide smart waste classification features in real time. Label detection and object localization has been combined to improve the classification accuracy, and supported by a reference database that maps detected objects to waste categories. Core features such as image scanning, object detection, and waste categorization were successfully integrated and tested. Testing confirmed that the application performs effectively under most conditions, with some limitations in image clarity and recognition accuracy. Feedback from users was used to improve usability and data coverage. Although several challenges have encountered, the system has been successfully implemented to deliver a functional and scalable solution.

CHAPTER 5: CONCLUSION

5.1 Introduction

This chapter provides a summary of the project's achievements of the project, identifies its limitations, suggest areas for future improvements, and concludes the development journey. The main objective of this project was to develop a user-friendly mobile application that assists users in identifying and categorizing waste items through image-based recognition. The application enables users to upload or capture photos of waste, which are then processed using the Google Vision API to determine the appropriate waste category. This chapter highlights the implemented features, the practical outcomes, and the broader potential of the system in promoting better waste management habits and environmental awareness through technology.

5.2 Project Achievements

This project has successfully fulfilled its main objectives, which focused on developing a mobile waste sorting application and enabling automatic classification through image input. Table 5.1 shows the achievement of the project's objectives.

Table 5.1: Achievement of Project Objective

Objectives	Achievements
To develop a waste sorting mobile application	A mobile app was successfully developed using React Native and Firebase, featuring user login, scanning, waste history, feedback, and admin management.
To allow users to upload photo of waste items for automatic identification and categorization	Integrated Google Vision API and Firebase Firestore to classify waste items based on scanned images. Labels are matched to stored objects and mapped to categories.

The app has been thoroughly tested using a variety of images, including common household waste, blurry photos, multi-object scenes, and items not found in the database. These tests helped verify the effectiveness and limitations of the system, and ensured that the app

remains stable and informative across different use cases. These achievements show that the application has successfully met its intended goals and provides a practical tool for encouraging proper waste disposal habits.

5.3 Limitations and Constraints

While the system met its primary goals, a number of limitations and technical constraints were encountered. During the development and testing process, several technical, functional, and resource-related challenges were encountered. These challenges impacted development speed, system accuracy, and cost-efficiency.

One major challenge was the inconsistency of the Google Vision API results. The API occasionally produced inaccurate or inconsistent results, especially when users submitted unclear, blurry, or poorly lit images. In some cases, it failed to recognize any objects, or it returned general or unrelated labels. This inconsistency affected the waste classification accuracy. Hence, a comprehensive object database had to be written manually, where common household objects were manually added to the Firestore database. This acted as a fallback when the API couldn't detect items. An administrative feature is introduced to manage the data efficiently to streamline ongoing updates. Due to these inconsistencies, extensive testing was conducted using different types of images, including various lighting conditions, backgrounds, and item arrangements to have a better understanding of the API's behaviour and to improve the accuracy of the system.

Another challenge was the inability of the API to detect the material of an item. For example, when a user scans a picture of a bottle, the Vision API might detect it as just a "bottle" without indicating whether it's made of plastic, glass, or metal. This makes accurate classification difficult, especially for items that look similar but belong to different waste categories. Thus, all possible descriptions for bottle were included in the "objects" collection

in the Firestore. This allowed the app to provide users with multiple possibilities along with appropriate disposal instructions for each type.

The system also struggled with multiple objects in a single image. The Vision API sometimes detected several items at once, which led to confusion when determining the correct item to classify. Hence, the app displays all detected labels that match entries in the "objects" collection. This means users may see multiple possible results and need to identify the most relevant one themselves. While this approach keeps the system simple, it may affect the accuracy of waste classification when images contain more than one item.

Google Vision API's free tier is limited to 1,000 units per month. Exceeding this limit incurs additional charges, which could be costly during testing or widespread usage. During testing or high usage periods, this restriction pose a challenge for maintaining consistent performance without escalating costs. Hence, mock data was used during development to simulate results without consuming real quota. This challenge was also mitigated through the use of Google Cloud student credits, which provided additional quota for testing and development.

Finally, from a user interface perspective, ensuring responsive design across different screen sizes and devices required additional effort. Some UI components, such as cards, images, and buttons, initially overflowed or broke layout on smaller devices, especially during image preview or result display. Styling adjustments and layout restructuring were implemented to improve compatibility.

5.4 Future Works

There are several areas where the application can be improved in future versions:

i. Replacing or supporting Google Vision with a custom-trained image classifier tailored to waste images could improve accuracy, especially for local or uncommon waste items.

ii. Adding logic to distinguish object materials more reliably, possibly by combining metadata or image texture analysis.

iii. Integrating local processing capabilities to enable scanning without internet access.

iv. Expanding the application to support multiple languages for broader accessibility.

v. Introducing points, badges, or rewards for the gamification to motivate users to consistently practice proper waste sorting.

5.5 Conclusion

In conclusion, this project successfully developed a mobile waste sorting application that allows users to scan and identify waste items using the Google Vision API. The main objective was to provide a tool that supports automatic waste classification and guides users on proper disposal. Throughout the development process, new technical skills were gained, particularly in integrating machine learning APIs, working with Firebase, and implementing real-time mobile features using React Native. The system currently serves as a functional prototype that can help educate the public on waste categorization and responsible disposal. By allowing users to scan items, view detection results, track their history, and even report inaccurate classifications, the application offers a user-friendly and educational experience. Despite certain limitations, the system could be scaled for broader use, offering better accuracy, personalization, and community involvement in waste management for future enhancements.

REFERENCES

Amal Abdel-raouf, Alaa Sheta, AbdelKarim Baareh, & Rausch, P. (September, 2024). *Barcode-less Fruits Classification Using Deep Learning.* IAES International Journal of Artificial Intelligence, 13(3), 3211–3211. https://doi.org/10.11591/ijai.v13.i3.pp3211-3217

Anil, R., Borgeaud, S., Alayrac, J.-B., Yu, J., Soricut, R., Schalkwyk, J., Dai, A. M., Hauth, A., Millican, K., Silver, D., Johnson, M., Antonoglou, I., Schrittwieser, J., Glaese, A., Chen, J., Pitler, E., Lillicrap, T., Lazaridou, A., & Firat, O. (December, 2023). *Gemini: A Family of Highly Capable Multimodal Models*. ArXiv.org. https://doi.org/10.48550/arXiv.2312.11805

Challa Kaushik Koundinya, Shaik Bhikku Saheb, Surya Sai Gopalam, Chakravarthula Abhiraman, & Dr. M. Madhusudhana Subramanyam. (September, 2024). *Agile Software Development.* African Journal of Biomedical Research. https://doi.org/10.53555/ajbr.v27i2s.1258

Daminion. (2023). *AI Labels vs AI Objects | Daminion.* Digital Asset Management Solution. https://daminion.net/docs/google-vision-labels-vs-objects/#:~:text=Unlike%20label%20detection%2C%20which%20provides,present%2C%20including%20their%20confidence%20scores.

Hafizah Hammad Ahmad Khan, Nabila Ahmad, Noorlailahsuna Mohd Yusof, Azyyati Anuar, Azlyn Ahmad Zawawi, & Shehu El-Rasheed (October, 2024). *Pathways to Sustainability: Empowering Indigenous Communities through Recycling Education*. Information Management and Business Review, 16(3(I)S), 1086–1093. https://doi.org/10.22610/imbr.v16i3(i)s.4175

International Trade Administration. (March, 2024). Malaysia Waste Management. International Trade Administratio. https://www.trade.gov/market-intelligence/malaysia-waste-management

Joshi, M. (2025). *React Native vs Flutter: What to Choose in 2025 | BrowserStack*. BrowserStack. https://www.browserstack.com/guide/flutter-vs-react-native

Karanrat Thammarak, Sirisathitkul, Yaowarat Sirisathitkul, Prateep Kongkla & Sarun Intakosum. (July, 2022). *Automated Data Digitization System for Vehicle Registration Certificates Using Google Cloud Vision API*. Civil Engineering Journal, 8(7), 1447–1458. https://doi.org/10.28991/cej-2022-08-07-09

Karrar Hameed Abdulkareem, Mohammed Ahmed Subhi, Mazin Abed Mohammed, Mayas Aljibawi, Nedoma, J., Martinek, R., Muhammet Deveci, Shang, W.-L., & Witold Pedrycz. (June, 2024). *A manifold intelligent decision system for fusion and benchmarking of deep waste-sorting models*. Engineering Applications of Artificial Intelligence. https://doi.org/10.1016/j.engappai.2024.107926

Muhammad Imran & Norah Almusharraf. (May, 2024). *Google Gemini as a next generation AI educational tool: a review of emerging educational technology*. Smart Learning Environments, 11(1). https://doi.org/10.1186/s40561-024-00310-z

Saptaputra, E. H., Nunnun Bonafix, & Araffanda, A. S. (April, 2023). *Mobile App as Digitalisation of Waste Sorting Management.* IOP Conference Series Earth and Environmental Science. https://doi.org/10.1088/1755-1315/1169/1/012007

Sekar, S. (2019). *Waste Classification data*. Kaggle.com. https://www.kaggle.com/datasets/techsash/waste-classification-data

Singh, S., P Sarala, Chandra, S. K., & Kumar, M. D. (2024). *DeepFake Image Detection and Classification using EfficientNet Model*. 1–5. https://doi.org/10.1109/otcon60325.2024.10687774

Sureka, S. (2022). Should Professionals Use Firebase? Pros and Cons | Pangea.ai. Pangea.ai. https://pangea.ai/resources/should-professionals-use-firebase-pros-and-cons

Tan, M., & Le, Q. V. (2019). *EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks*. ArXiv.org. https://arxiv.org/abs/1905.11946

Wan Md Syukri Wan Mohamad Ali, Zaipul Anwar Zainu, Nooh Abu Bakar, & Ahmad Rahman Songip. (November, 2015). *SUSTAINABLE HAZARDOUS WASTE MANAGEMENT IN MALAYSIA.*

APPENDIX

```
const analyzeImage = async (base64Image, photoURL) => {
        setAnalyzeLoading(true);

        try {
            const response = await axios.post(
                `https://vision.googleapis.com/v1/images:annotate?key=${GOOGLE_VISION_API_KEY}`,
                {
                    requests: [
                        {
                            image: { content: base64Image },
                            features: [
                                { type: "OBJECT_LOCALIZATION", maxResults: 10 },
                                { type: "LABEL_DETECTION", maxResults: 10 }
                            ]
                        }
                    ]
                }
            );

            const objectAnnotations = response.data.responses[0].localizedObjectAnnotations || [];
            const labelAnnotations = response.data.responses[0].labelAnnotations || [];

            const objectMap = new Map();

            objectAnnotations.forEach(obj => {
                objectMap.set(obj.name.toLowerCase(), {
                    ...obj,
                    detectionType: 'object'
                });
            });

            labelAnnotations.forEach(label => {
                const key = label.description.toLowerCase();
                if (!objectMap.has(key)) {
                    objectMap.set(key, {
                        name: label.description,
                        score: label.score,
                        boundingPoly: { normalizedVertices: [] },
                        detectionType: 'label'
                    });
                }
            });

            const combinedRawObjects = Array.from(objectMap.values());

            const cleanedObjects = [];

            for (const obj of combinedRawObjects) {
                const { name, score, boundingPoly, type } = obj;

                const objectQuery = query(collection(firestore, 'objects'), where('objName', '==', name));
                const objectSnapshot = await getDocs(objectQuery);

                if (objectSnapshot.empty) continue;

                const objectDoc = objectSnapshot.docs[0].data();
                const categoryIds = objectDoc.categoryId;
                const categoryDescs = objectDoc.categoryDesc;

                const categoryQuery = query(collection(firestore, 'categories'), where('categoryId', 'in', categoryIds));
                const categorySnapshot = await getDocs(categoryQuery);

                const categories = categorySnapshot.empty ? [] : categorySnapshot.docs.map(doc => doc.data());
                const finalCategoryDesc = categoryDescs || ['No description'];

                cleanedObjects.push({
                    name,
                    score,
                    vertices: boundingPoly.normalizedVertices || [],
                    categories: categories.map(c => c.categoryName) || ['Unknown'],
                    categoryDesc: finalCategoryDesc,
                    recycleInstruction: categories.map(c => c.categoryRecycle) || ['N/A'],
                    isRecyclable: categories?.[0]?.isRecyclable ?? false,
                    photoURL,
                    detectionType: type,
                });
            }

            setDetectedObjects(cleanedObjects);
            setIsDrawerVisible(true);
        } catch (error) {
            Toast.show({ type: 'error', text1: 'There was a problem analyzing the image.', text2: error.message || 'Please try again later.' });
        } finally {
            setAnalyzeLoading(false);
        }
    };

```