Trash to Treasure: AI-based Trash Classification and Recycling Rewards Platform

Project Team

Ayesha Eman 20I-0609
Imama Amjad 20I-1819
Manahil Shakeel 20I-2302

Session 2020-2024

Supervised by

Dr. Akhtar Jamil

Department of Computer Science

National University of Computer and Emerging Sciences
Islamabad, Pakistan

June, 2024

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This report outlines the requirements and specifications for "Trash to Treasure," an innovative AI-based trash classification and recycling rewards platform. The primary objective of this platform is to address the pressing issue of subpar waste disposal practices and low recycling rates by implementing an efficient system for identifying and segregating various types of waste materials (single-use plastics, paper, metal cans for beverages, and food items), within organizational settings such as workplaces, universities, schools, and hospitals. By incentivizing recycling through a rewards system, Trash to Treasure aims to promote sustainable waste disposal practices and contribute to environmental conservation efforts.

The intended audience for this document includes developers, project managers, marketing staff, users, testers, and documentation writers. Developers will rely on this document to understand the functional and non-functional requirements of the system, as well as its architectural and data design. Project managers will utilize this report to oversee the development process, allocate resources effectively, and ensure that the project meets its objectives within the specified timeline. Marketing staff will gain insights into the platform's features and capabilities to devise strategies for promoting its adoption and usage among target users. Users will refer to this document to familiarize themselves with the functionality and benefits of Trash to Treasure. Testers will rely on the specified use-case scenarios to validate the system's behavior and functionality. Additionally, documentation writers will use this report as a reference to create user manuals and technical documentation for end-users and developers.

1.1 Existing Solutions

In evaluating existing smart waste management solutions, it is essential to critically assess their strengths and weaknesses to identify areas where "Trash to Treasure" can offer

improvements. Below is an analysis of four notable systems:

1.1.1 Bin-e

Bin-e [2] is an AI-powered smart waste bin designed for public and commercial spaces, offering automated sorting and waste compaction to improve recycling efficiency.

Strengths:

- *Automatic Sorting and Compression:* Bin-e utilizes AI to automatically sort waste into appropriate categories and compress plastics and paper, enhancing storage efficiency [2].
- *Cloud Connectivity:* The system monitors fill levels and provides real-time data analytics, facilitating efficient waste management.
- *Customizable Design:* The system supports branding, color customization, and has a big built-in screen for content management, enhancing user engagement and adaptability.
- *Notifications and Dashboard:* Provides notifications for fullness and a reporting dashboard for detailed analytics.

Weaknesses:

- *Geographical Constraints:* Based in Poland, Bin-e's service and support are limited in other regions, including Pakistan, affecting scalability and accessibility.
- *Limited Sorting Options:* The system does not support sorting of organic or contaminated items, limiting its usability in diverse waste streams.
- *No Incentive Mechanism:* Lacks gamification or a rewards system, which are crucial for encouraging sustainable behavior.

1.1.2 TrashBot

TrashBot [4], developed by CleanRobotics, is a highly accurate AI-powered waste sorting system designed for high-traffic areas.

Strengths:

- *High Sorting Accuracy:* TrashBot achieves a sorting accuracy of approximately 96%, significantly reducing contamination [3].

- *Customizable Waste Streams:* Offers customizable configurations for 2, 3, or 4 separate waste streams to suit specific requirements.
- *Dynamic User Engagement:* Features a big screen with dynamic content, a light-up logo, and notifications for enhanced user interaction.
- *Cloud Connectivity and Analytics:* Provides a reporting and analytics dashboard with cloud storage, ensuring efficient waste management.
- *Capacity:* Boasts a total capacity of 80 gallons (304 liters) for handling large volumes of waste.

Weaknesses:

- *Cost and Maintenance:* The advanced technology and service costs may hinder adoption, especially in regions like Pakistan, where budgets for waste management are limited.
- *Slow Cycle Time:* Requires waste to be disposed of one item at a time, with a cycle time of 4-6 seconds, which can extend to 7-12 seconds during heavy usage.
- *No Rewards Mechanism:* Does not include gamification or user incentive features.

1.1.3 EvoEco

EvoEco [7] is a modular waste management system designed for businesses to enhance sustainability and optimize waste disposal costs.

Strengths:

- *Responsive Video Content:* Delivers customizable video messages to educate users about proper waste disposal.
- *Analytics Dashboard:* Provides detailed insights into waste disposal behavior, enabling businesses to make data-driven decisions.
- *Compact Design:* Its Seattle-based design focuses on efficient use of space with a 23-gallon capacity.

Weaknesses:

- *Absence of Automatic Sorting:* Relies on users for waste segregation, increasing the likelihood of errors.

- *Additional Costs:* Customization of video content requires extra expense, limiting adoption in cost-sensitive regions like Pakistan.
- *No Rewards System:* Does not offer a points-based incentive mechanism to engage users.

1.1.4 Oscar Sort

Oscar Sort [9] is a smart recycling assistant designed to guide users in proper waste disposal with real-time feedback and retrofitting capabilities.

Strengths:

- *Interactive Feedback and Gamification:* Oscar Sort greets users through its Discovery Mode, visually demonstrating proper waste disposal methods. It reinforces correct actions with positive feedback and uses humorous responses for incorrect disposal, making the process engaging and educational.
- *Retrofits to Existing Bins:* Designed to attach to existing bins, offering flexibility and cost savings.
- *Advanced Notifications:* Provides fullness notifications and predictive trends to optimize waste collection schedules.
- *Analytics Dashboard:* Includes reporting features to track and analyze waste disposal patterns.

Weaknesses:

- *No Automatic Sorting:* Requires users to manually sort waste, increasing the likelihood of contamination and errors/cheating.
- *Geographical Constraints:* Based in Vancouver, Canada, its availability and support in regions like Pakistan are limited.
- *No Direct Points-Based System:* Unlike systems designed with a built-in points or gamification mechanism for ongoing user engagement, Oscar Sort relies on external reward schemes, which may not always be available or scalable. [6]
- *Dependence on Venue-specific Rewards:* While Oscar Sort offers rewards in some installations, the incentive systems vary widely by location and are not universally implemented, limiting consistent user engagement across all deployments [10].

1.1.5 MyMatR

MyMatR [5] is a smart waste bin designed to sort waste at the source of disposal automatically. MyMatR ensures efficient waste categorization into landfill and recycling bins and offers a dashboard for real-time data reporting, making waste management more sustainable and streamlined.

Strengths:

- *Automatic Sorting:* MyMatR incorporates artificial intelligence for waste sorting at the point of disposal, ensuring efficient categorization into landfill and recycling bins.
- *Real-Time Data Reporting:* The system provides Wi-Fi or 4G/LTE connectivity, enabling users to access real-time data about their waste disposal habits and export this data for analysis.
- *Fullness Detection:* Offers fullness detection capabilities, sending alerts when bins reach capacity, optimizing waste collection efficiency.
- *Educational Display:* Features a 7-inch embedded screen that educates users on proper waste disposal practices.
- *Customization Options:* Provides options for container shapes, sizes, branding, and even advertising, making it suitable for businesses and varied use cases.

Weaknesses:

- *Limited Waste Streams:* Only supports two internal waste streams, limiting flexibility in complex waste scenarios.
- *No Rewards Mechanism:* Does not include gamification or user incentive features.
- *Sorting Accuracy Unknown:* The system's sorting accuracy is not disclosed, making it difficult to benchmark against competitors.
- *Binary Decision Limitation:* Limited to a binary decision process, which may not adequately address diverse waste categorization needs.
- *Geographical Constraints:* Availability and support in regions like Pakistan are unclear, potentially limiting accessibility and scalability.

Table 1.1: Comparison of Existing Solutions

System Name	System Overview	System Limitations
Bin-e [2]	AI-powered smart waste bin for automated sorting and waste compaction. Features include cloud connectivity, notifications for fullness, and a customizable design with branding options.	Limited sorting options (no support for organic waste), lacks incentive mechanisms, and geographical constraints, particularly in regions like Pakistan.
TrashBot [4]	AI-powered waste sorting system designed for high-traffic areas. Offers high sorting accuracy (96%), dynamic user engagement through a big screen, customizable waste streams, and analytics dashboard.	High costs, slow cycle time for disposal (up to 7-12 seconds), and no direct rewards or gamification features.
EvoEco [7]	Modular waste management system focused on educating users with video content, providing analytics for sustainability, and compact design for smaller spaces.	No automatic sorting, additional costs for video customization, and absence of a points-based rewards system.
Oscar Sort [9]	Smart recycling assistant with Discovery Mode that provides real-time feedback and gamification. Retrofits to existing bins and offers analytics and fullness notifications.	No automatic sorting, dependence on venue-specific rewards, lack of a universal points system, and limited availability in regions like Pakistan.
MyMatR [5]	Smart waste bin with AI-based sorting, real-time data reporting, and fullness detection. Features include a 7-inch educational display and customizable container designs.	Limited to two waste streams, sorting accuracy is not disclosed, lacks a rewards mechanism, and binary decision limitations.
Trash to Treasure	AI-based platform that combines waste sorting and gamified rewards to encourage recycling. Offers user-friendly interfaces, a points-based incentive system, and scalability for organizational settings.	Being developed; requires validation for sorting accuracy and implementation in diverse regions, including Pakistan.

Table 1.2: Feature-based Comparison of Smart Waste Management Systems

Feature	Bin-e	TrashBot	EvoEco	Oscar Sort	MyMatR	Trash to Treasure
AI-based Sorting	Yes	Yes	No	Yes	Yes	Yes
Automatic Sorting	Yes	Yes	No	No	Yes	Yes
Compression of Recyclables	Yes	No	No	No	No	No
Points and Rewards System	No	No	No	No	No	Yes
User Engagement through Gamification	No	No	No	Yes	No	Yes
Customizable Interface for Users	No	Yes	Yes	Yes	Yes	Yes
Web and Mobile Application for Different Users	No	No	No	No	No	Yes

1.2 Problem Statement

The "Trash to Treasure" initiative addresses the critical challenge of inadequate waste management and low recycling rates in Pakistan by leveraging AI to classify and segregate waste within institutions, promoting environmental sustainability through an incentivized recycling rewards system.

1.3 Background and Need for the Solution

The development of the "Trash to Treasure" system stems from a critical need to address the pervasive issue of inadequate waste management practices and low recycling rates, particularly prevalent in Pakistan. Pakistan generates approximately 49.6 million tons of

solid waste a year, which has been increasing more than 2.4 percent annually. [14]. Pakistan is the second-largest domestic market for plastic, with a meagre 18 percent recycling potential and only 3 percent of plastic being recycled locally. [13]. [8]. Despite efforts to promote environmental awareness, there exists a significant lack of motivation among individuals and organizational settings to actively engage in recycling and sustainable waste disposal practices. This reluctance is exacerbated by the commonplace practice of littering and the widespread contamination of recyclable materials with non-recyclable ones, leading to the generation of mixed waste. Mixed waste causes problems later on as it requires significant energy and resources for segregation. [11]. The prevailing preference for plastic products, combined with misconceptions about individual responsibility and the allure of material acquisition, further complicates matters. [12]. In Pakistan, where even educated populations struggle with low recycling rates, the challenge of managing mixed waste is particularly daunting.

Our system targets specific waste materials such as plastic, metal, paper, and food items, which are commonly generated in institutions like schools, universities, workplaces, and hospitals. These materials hold immense potential for recycling, yet they often end up as mixed waste due to various factors including a lack of awareness, inadequate infrastructure, and the absence of a rewarding and user-friendly system. The "Trash to Treasure" platform aims to mitigate these challenges by implementing an innovative AI-based trash classification and recycling rewards system. By efficiently identifying and segregating different types of waste materials within organizational settings, the system incentivizes recycling through a rewards mechanism, thereby promoting sustainable waste disposal practices and contributing to environmental conservation efforts. Through the implementation of this system, we seek to elevate recycling rates, reduce strain on environmental resources, and foster a culture of responsible waste management within institutions.

1.4 Scope

1.4.1 Project Scope and Focus

Our project consists of an AI-based Trash Classification Module, a Smart Bin, a Client Mobile App, a Client Web App and an Admin Web App.

The AI-based Trash Classification module will be able to accurately differentiate between plastic, metal, and paper through computer vision. This module will generate unique QR codes associated with identified waste types, storing relevant details such as the type of trash, timestamp, expiry time, number of items (if same type disposed), associated PIN Code, status (redeemed or not redeemed), weight (for paper only) and the corresponding points awarded in a secure database.

We will also develop and assemble a Smart Bin equipped with Raspberry Pi technology to facilitate efficient waste disposal and recycling efforts. The Smart Bin will feature four smaller bins designated for plastic, paper, metal, and mixed waste. It will have a small door under the platform. A servo motor will be attached to the door and this door will only open after trash has been correctly identified. The Smart Bin will also have a second slide operated by a servo motor and it will be pointing to the smaller bins.

User will dispose of the trash item in the bin. The trash item will slide onto a platform beneath a mounted camera, preventing retrieval after disposal. The camera, integrated with image processing capabilities, will call on the AI-based Trash Classification module. After successful classification, the second slide will rotate to the opening of the smaller bin which fits the correctly identified category of trash. The small door on platform will open and trash item will slide to the designated bin. If trash item does not belong to the categories within our scope or if more than 1 category of trash is thrown together, our system will consider it as mixed waste and no points will be awarded.

For this project, we have assigned points to different waste categories as follows:

- Plastic: 70 points
- Metal: 50 points
- Paper: 30 points

Each point earned by the user corresponds to 1 rupee, providing a straightforward incentive system for recycling efforts. A user needs to have minimum 1000 points to be able to use them to receive discounts. Points can be changed by the admin through use of Admin Web App.

The Client Mobile App module, designed for users, will facilitate easy point collection, profile management, history tracking, and point redemption, reinforcing positive recycling behavior. The Client Mobile App will allow user to complete challenges and earn badges based on how much they have recycled. The user can use the app to scan the QR code generated by the AI-based Trash Classification Module which will be displayed on a screen. User can still use the QR code even if they do not have the app by noting down the code or taking a photo of the code displayed on the screen. Through use of Client Mobile App, user can also give feedback regarding our system.

The Client Web App will facilitate registration for shops, integration with the point system, and updating the status of redeemed points in the database. It will allow cashiers to scan the QR code from user's client mobile app and give discounts based on earned points. This module will also allow cashiers to manually input discount code that user will give them and validate if it right or not. Discounts are only given for valid, collected points.

An Admin Web Module will serve as an administrative interface for our system. It will allow the admin to manage points system, manage the integrated stores,manage user feedback, and manage user accounts.

1.4.2 Exclusions and Limitations

Our scope is limited to trash that is commonly found/produced in an organization such as FAST University. This type of trash includes paper, card, paper cups, Styrofoam cups and boxes, plastic straws, plastic plates, plastic cups, plastic cutlery, aluminum soda cans, etc. The system should not detect any other material outside of these classes and should not reward any points if the item does not belong to one of our considered classes. There is no restriction on the vicinity of partner shops and cafes; however, they must be registered through our system. There is also no restriction on type of business registered through the system that will be offering discounts.

We have excluded functionality related to order management and delivery management from the project scope. The Client Web App module will focus solely on scanning QR codes and updating their status in the database, without managing inventory, managing order, managing delivery, or generating receipts.

Additionally, the system will not involve hardware for manual or automated waste segregation beyond the Smart Bin's capabilities, and it will not dictate the handling of classified waste after it has gone into the smaller bins. A basket has been used to showcase the capabilities of the platform under the camera. No motion sensor is used. Glass will no be classified as 'glass' as we do not have the required heat sensor, so the system can at times classify glass as 'plastic' because of lack of differences between two. Moreover, no humidity sensor is used to detect wetness of paper, since it is not feasible to clean the humidity sensor in run-time environments after it becomes wet once. If it is not cleaned, it will cause a defective output. That is why we will not be concerned about the wetness of paper. Furthermore, instead of OLED screen, we will be showcasing the functionality of the screen through our laptop due to budget constraints.

Due to budgetary and feasibility constraints, the outer enclosure or boundary structure of the bin has not been developed as part of this project. Although the internal skeleton of the bin has been constructed, including its functional components, the main door mechanism that would close after trash is placed has not been implemented. This limitation focuses the project on the core functionality of the system, while allowing enhancements to the physical structure for future development.

1.5 Modules

1.5.1 Data Collection and Pre-processing

1. Datasets from Kaggle will be used. [1]

2. Additional data will be collected by taking samples from the university cafeteria.

3. The collected data will be prepared for training by pre-processing the images.

4. This may include resizing images to a consistent resolution, normalization (e.g., scaling pixel values to a specific range), and augmentation techniques (e.g., rotation, flipping, cropping) to increase the dataset's diversity and help the model generalise better.

5. Extracting relevant features from the images that are essential for the classification task.

6. After processing, the object classification model will be trained using the pre-processed dataset.

1.5.2 AI-based Trash Classification Module

The pivotal module in our project. Our AI model will classify and categorize types of trash and reward the user with points that they can redeem later.

1. Utilising the model we trained for object detection to identify and classify materials into categories like plastic, metal, and paper.

2. If object is not in any of the above categories, it might still be identified but no points/rewards will be given.

3. If there are multiple objects belonging to the same accepted category of trash, the reward will be multiplied by number of items in case of plastic and metal. In case of paper, weight of the trash item will be considered and used for awarding the points.

4. If there are multiple objects each belonging to different categories of trash or belonging to categories of trash that our system does not cater, they might still be identified but will be classified as mixed waste and no points/reward will be given.

1.5.3 Smart Bin

The essential hardware module in our project. Our Smart Bin will utilise Raspberry Pi technology. It will be assembled using smaller plastic pins, camera, plastic slides, weight sensors, stepper motors, servo motors, OLED screen and wires.

1. When user will throw trash, it will slide onto a platform so user cannot retrieve it again.

2. Camera will be fixed above the platform. It will be connected to our AI-based Trash Classification Module.

3. After trash item has been classified, our main slide will rotate to the opening of the smaller bin which fits the category of the trash. It will use a stepper motor for this function. Weight can also be noted at this point.

4. A small door will open from the platform with the help of a servo motor. The trash item will fall onto the slide and descend into the correct smaller bin.

5. If object is not in any of the above categories, it might still be identified but no points/rewards will be given.

6. If there are multiple objects belonging to the same accepted category of trash, the reward will be multiplied by number of items in case of plastic and metal. In case of paper, weight of the trash item will be considered and used for awarding the points.

7. If there are multiple objects belonging to different categories of trash, it will be considered mixed waste and no points/rewards will be given.

8. After the trash item has been classified, a QR code and a PIN Code will be generated containing relevant information such as the category of the trash and additional details required for tracking or rewards.

9. The QR code and the PIN Code will both be displayed on the screen attached to the Smart Bin.

10. The user can scan the QR code using their mobile app to collect points.

11. Alternatively, the user can manually enter the PIN Code into their mobile app to collect points or note down the PIN Code on paper,etc. for later use.

12. The details of the classified trash item along with the generated QR code and PIN Code will be stored in the database for tracking and reward purposes.

1.5.4 Client Mobile App

This module acts as the primary interface between users and our trash classification and rewards system, shaping their experience, fostering engagement, and ultimately driving positive behavioural change towards sustainable waste practices. It serves as the bridge that transforms recycling from a mundane task into a rewarding and enjoyable journey.

1.5.4.1 User Registration and Profile Management

1. Creating and managing user profiles. This will allow users to set up profiles, track their recycling history, and customize their app preferences.

1.5.4.2 QR Code Scanning

1. QR code scanning functionality will allow users to scan QR codes associated with different types of trash.

1.5.4.3 PIN Code Entry

1. Alternatively, users can enter a PIN code displayed on the Smart Bin's screen if they prefer not to use the QR code scanner. This PIN can also be manually noted for later entry if the user does not have the app available at the moment.

1.5.4.4 Real-Time Point Tracking

1. Real-time point tracking for users will show the current points earned through recycling activities. The user needs to have minimum 1000 points available to be eligible to use them for redemption.

1.5.4.5 Recycling Activity History

1. Maintaining a history of user recycling activities, including dates of recycling activities, types of materials recycled, points earned, and rewards redeemed.

1.5.4.6 Gamification Elements

1. Introduction of challenges, badges, or a point-based leveling system to enhance user engagement and promote consistent recycling.

1.5.4.7 Feedback Module

1. Users of the application can send their feedback to the administration of our Trash to Treasure system.

1.5.4.8 View Integrated Stores

1. Users can access detailed information about partnered stores and cafes, enabling them to identify locations where they can redeem their rewards.

1.5.5 Client Web App

The web interface is used by partnered shops and cafes that utilise the points collected by users and offer them discounts on purchases and orders.

1.5.5.1 Shop and Cafe Registration

1. Allowing shops and cafes to easily register on the platform.

1.5.5.2 QR Code Scanning

1. QR code scanning functionality will allow cashiers to scan QR codes provided by customers to make sure they have not been redeemed already and/or expired.

1.5.5.3 PIN Code Entry and Authentication

1. Cashiers can enter the PIN code provided by customers to verify and update the status of rewards directly if the QR code feature is not used.

1.5.5.4 Integration with Point System

1. Integrating the web interface with the point system for seamless transactions. This is to update status of redeemed points in the database only.

1.5.6 Admin Web App

The Admin Web Module is indispensable for overseeing and managing various aspects of the trash classification and recycling rewards platform. It serves as the control center for ensuring the platform's smooth operation, security, and compliance.

1.5.6.1 User Account Management

1. Providing an interface for admins to manage user accounts and access.

1.5.6.2 Point System Management

1. Allowing admins to manage the overall point system. Adjust point values, set rewards, and monitor point distribution.

1.5.6.3 Shop and Cafe Management

1. Creating a console for admins to manage registered shops and cafes. They have to approve registrations and ensure compliance with the platform's standards.

1.5.6.4 Feedback Management

1. Developing a module for receiving and managing user feedback.

1.6 User Classes and Characteristics

We identify that three types of users will use our system: Recyclist, Cashier and Admin.

Their characteristics are explained below.

Table 1.3: User Classes and Characteristics in Trash to Treasure

User Class	Description
Recyclist	A Recyclist is environmentally conscious, comprising students, professionals, and eco-aware individuals. They engage in recycling via mobile apps, scanning QR codes for rewards. Active participants, they seek convenience and seamless experiences, often embracing gamification elements for engagement.
Cashier	A Cashier is a shop or cafe employee on the Trash to Treasure platform responsible for QR code scanning and point redemption. They use the client web app for processing transactions, communicating with customers, and collaborating with administrators. They require a user-friendly interface for efficient point validation and access to real-time information.
Admin	Administrators manage Trash to Treasure operations. They use web-based tools for tasks like user management and system monitoring. They require comprehensive data analytics, user-friendly interfaces, and tools for addressing user inquiries and managing system configurations.

1.7 Work Division

Table 1.4: Work Division Among Project Members

Member Name	**Assigned Tasks**	**Deliverables**
i20-0609 Ayesha Eman	Data Collection, Preprocessing. Model Training, Testing and Fine-tuning.	Data preprocessing module, trained deep learning models (ResNet, EfficientNet, YOLOv8, Custom CNN), and comprehensive testing and performance evaluation metrics.
i20-1819 Imama Amjad	Mobile and Web App UI Design and Implementation in Flutter. Project documentation.	Fully functional web applications and mobile application. Front-end interface design. Project report
i20-2302 Manahil Shakeel	Hardware Integration and Testing. Project documentation.	Fully integrated hardware-software system. Project report.

Chapter 2

Project Requirements

2.1 Use Case

2.1.1 Use Case Diagram

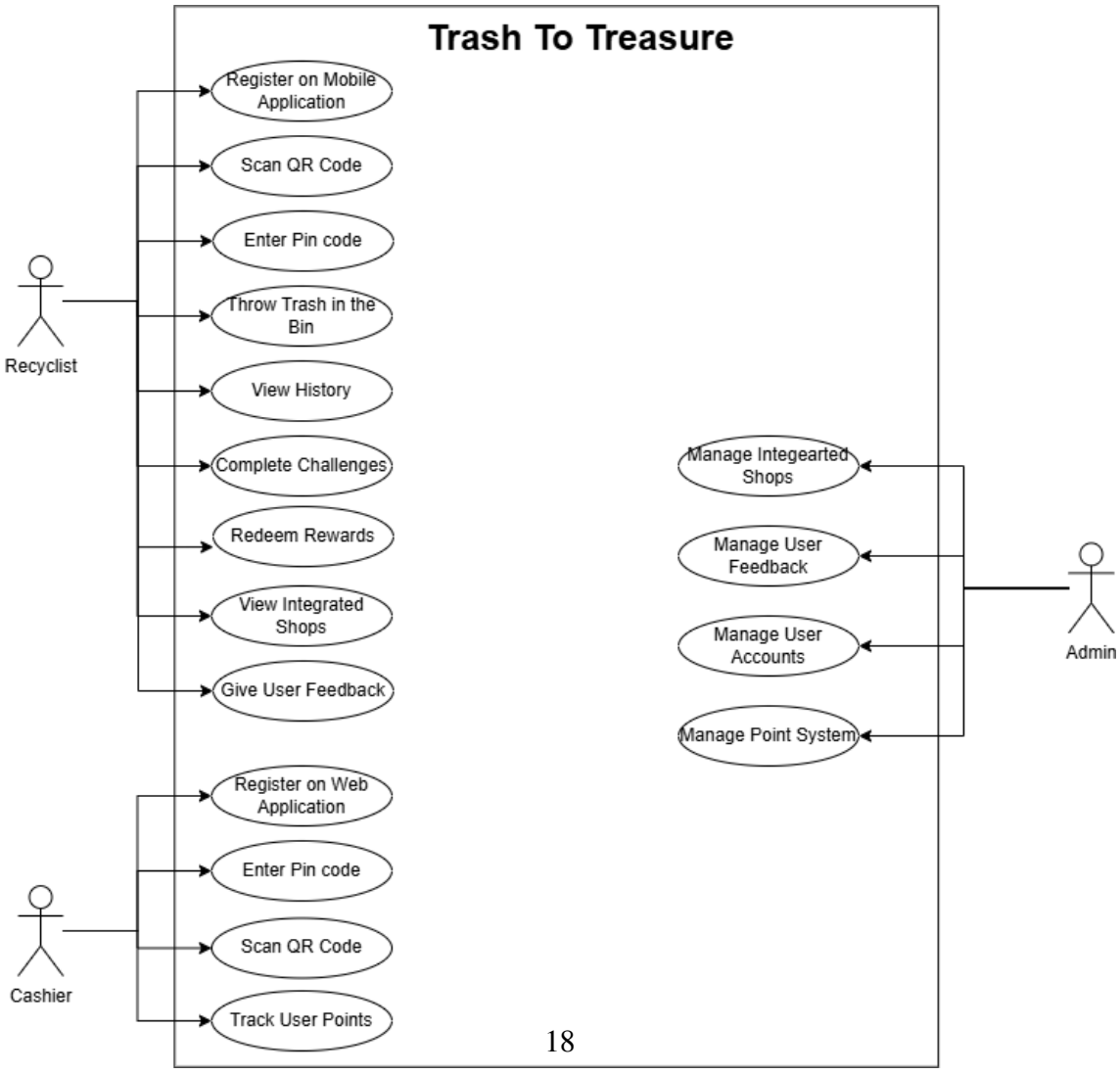


Figure 2.1: Use Case Diagram

2.1.2 Use Cases

1. Register on Mobile Application

Use Case ID	UC-1
Use Case Name	Register on Mobile Application
Actor	Recyclist
Type	Primary
Description	Users can create their accounts in the mobile application, set up, and manage their profiles as required

2. Scan QR Code

Use Case ID	UC-2
Use Case Name	Scan QR Code
Actor	Recyclist
Type	Primary
Description	User can scan the QR code displayed on screen via the mobile app scanning feature.

3. Enter Pin Code

Use Case ID	UC-3
Use Case Name	Enter Pin Code
Actor	Recyclist
Type	Primary
Description	User enters the pin code displayed on the screen into mobile app.

4. Throw Trash in the Bin

Use Case ID	UC-4
Use Case Name	Throw Trash in the Bin
Actor	Recyclist
Type	Primary
Description	User throws the trash into the bin, where it is categorized further inside the Bin.

5. View History

Use Case ID	UC-5
Use Case Name	View History
Actor	Recyclist
Type	Primary
Description	Users can view the history of redeemed discounts by them and their recycling history.

6. Complete Challenges

Use Case ID	UC-6
Use Case Name	Complete Challenges
Actor	Recyclist
Type	Primary
Description	User completes different challenges shown in the mobile app and earn rewards, increasing user consistency.

7. Redeem Rewards

Use Case ID	UC-7
Use Case Name	Redeem Rewards
Actor	Recyclist
Type	Primary
Description	Users can redeem their discounts on different items from the partnered shops according to the point system defined by the administration.

8. View Integrated Shops

Use Case ID	UC-8
Use Case Name	View Integrated Shops
Actor	Recyclist
Type	Primary
Description	Users can view the partnered shops with our system where the users can easily redeem their rewards and avail discounts.

9. Give User Feedback

Use Case ID	UC-9
Use Case Name	Give User Feedback
Actor	Recyclist
Type	Primary
Description	Users can give their feedbacks and add comments in the app which helps the app developers to make further improvements

10. Register on Web Application

Use Case ID	UC-10
Use Case Name	Register on Web Application
Actor	Cashier
Type	Primary
Description	Cashiers can register themselves on the platform using the web application module.

11. Scan QR Code (for web app)

Use Case ID	UC-11
Use Case Name	Scan QR Code
Actor	Cashier
Type	Primary
Description	Cashiers can scan the QR code using web app so that they can track the user's points and provide them discounts accordingly.

12. Enter Pin code (for web app)

Use Case ID	UC-12
Use Case Name	Enter Pin Code
Actor	Cashier
Type	Primary
Description	Cashiers enters the pin code provided by user in web app so that they can view and track user's points and provide them discounts accordingly.

13. Track User Points

Use Case ID	UC-13
Use Case Name	Track User Points
Actor	Cashier
Type	Primary
Description	Cashiers can track user's points (earned and deducted) and provide them calculated discounts.

14. Manage User Accounts

Use Case ID	UC-14
Use Case Name	Manage User Accounts
Actor	Admin
Type	Secondary
Description	Admins can create accounts for any users, they can delete the account if required, to ensure safe use of the system.

15. Manage Integrated Shops

Use Case ID	UC-15
Use Case Name	Manage Integrated Shops
Actor	Admin
Type	Secondary
Description	Admins can create the accounts of the integrated shops and can also delete any of them.

16. Manage Point System

Use Case ID	UC-16
Use Case Name	Manage Point System
Actor	Admin
Type	Secondary
Description	Admins can manage the points system as they can set values according to their needs. They can set the threshold, when reached, the user can avail discounts, they can set different values for different categories of trash i.e., plastic, metal and paper.

17. Manage User Feedback

Use Case ID	UC-17
Use Case Name	Manage User Feedback
Actor	Admin
Type	Secondary
Description	They can view user feedback, can delete any feedback given if regarded as inappropriate

2.1.3 Expanded Use Cases

1. Register on Mobile Application

    - **Main Success Scenario**

Actor Action	System Response
1. User clicks the register button.	
	2. System asks the user to create a new account or login with an existing account.
3. User chooses to create a new account option	
	4. System asks the user to enter his/her full name, email address, mobile number, and a new password for the account
5. User enters his/her details then clicks next button	
	6. System sends an OTP to the user's mobile phone and asks to enter the OTP for verification of the account created.
7. User enters the OTP	
	8. System verifies the user and shows a message saying account created successfully

    - **Actor**
      Recyclist
    - **Scope**
      Trash To Treasure System
    - **Alternate Flows**
      2a. If the account is already created user chooses the login option
      2b. System asks the user to enter the email address and password.
      2c. User enters the email address and password and is logged in to the system.
    - **Extensions**
      If at any time the system fails, user restarts the mobile application
    - **Pre-Condition**
      The user must have mobile application installed on their device and an active internet connection.
    - **Post-Condition**
      The user account is created successfully

2. Scan QR Code

- **Main Success Scenario**

Actor Action	**System Response**
1. User clicks the 'Scan QR Code' option from the side	
	2. System displays a screen with camera open to scan the QR code displayed on the trash bin screen.
3. User scans the QR code.	
	4. The mobile application saves that QR code and the pin code associated with it for later use by the user for rewards redemption

- **Actor**
  Recyclist
- **Scope**
  Trash To Treasure System
- **Extensions**
  a. If at any time the QR code scanning process generates an error the user tries again
  b. If at anytime the screen goes down or any issues exists with user's device, they can enter the pin code instead.
- **Pre-Condition**
  a. The user must be logged in to the system.
  b. The user's device's camera must be in working condition and necessary camera permissions must be allowed..
- **Post-Condition**
  The QR code is scanned successfully and stored for later use along with its associated pin code.

3. Enter Pin Code

- **Main Success Scenario**

Actor Action	System Response
1. User clicks the 'Enter Pin Code option from the side.	
	2. System displays a screen which asks the user to enter the pin code displayed on the trash bin screen.
3. User Enters that pin code manually into the mobile app.	
	4. The mobile application saves that pin code along with its associated QR code for later use by the user for rewards redemption

- **Actor**
  Recyclist
- **Scope**
  Trash To Treasure System
- **Extensions**
  a. If the screen says invalid pin code user tries to enter the pin code again.
  b. If anytime the screen goes down the user goes for the Scanning QR code option instead.
- **Pre-Condition**
  The user must be logged in to the system.
- **Post-Condition**
  The pin code is successfully entered in the system and stored for later use along with its associated QR code.

4. Throw Trash in Bin

- **Main Success Scenario**

Actor Action	System Response
1. User throws the trash into the opening of the bin.	
	2. System identifies the type of trash and classify in which relevant bin the trash will go to.
	3. The system will further dispose off the trash into the relevant bin.

- **Actor**
  Recyclist

- **Scope**
  Trash To Treasure System
- **Alternate Flows**
  1a. If the user does not throws trash into the bin, no QR code will be displayed on the display screen of the Trash Bin.
- **Pre-Condition**
  The user have some trash to throw into the bin from the catered categories i.e. plastic, paper, metal.
- **Post-Condition**
  The user has thrown trash into the respective opening of the bin

5. View History

- **Main Success Scenario**

Actor Action	**System Response**
1. User clicks the view history button from the side bar menu.	
	2. System displays the list of items user has recycled since registered to the platform.
	3. User clicks any one of the item in the list
	4. The system displays further information about the recycle made which includes the date, time, type of material recycled and the number of points earned through this recycle

- **Actor**
  Recyclist
- **Scope**
  Trash To Treasure System
- **Extensions**
  If the system fails to show the history, the user must restart the application
- **Pre-Condition**
  The user must be logged in the system and must have stable internet connection.
- **Post-Conditions**
  The history list is successfully displayed which shows the recycling history since the user has registered

6. Complete Challenges

   - **Main Success Scenario**

Actor Action	System Response
	1. The mobile app shows different challenges to the user.
2. The user completes the challenges	
	3. The system rewards badges to the user
4. The users compete against a leveling system	
	3. The system increases user level accordingly

   - **Actor**
     Recyclist
   - **Scope**
     Trash To Treasure System
   - **Extensions**
     If the level not increased or the badge not awarded when reached a pre-defined threshold, the user must refresh the page or restart the mobile application.
   - **Pre-Condition**
     The user must be a registered user and must have a stable internet connection to participate in different challenges.
   - **Post-Condition**
     The user is rewarded with the pre-defined reward, the badges are given and the user's level is increased respectively.

7. Redeem Rewards

   - **Main Success Scenario**

Actor Action	System Response
1. The user buys something from one of the partnered shops	
	2. According to the point system defined by admin and user's the QR code information, the user's points are used which confirms the user's purchase from the shop
	3. The system then maintains user's recycling history

   - **Actor**
     Recyclist

- **Scope**
  Trash To Treasure System
- **Alternative Flow**
  2a. If the user chooses to present code combination instead of the QR code to the cashier, then the points stored in that code combination are used for the purchase.
- **Extensions**
  If the user is unable to redeem their rewards from the partnered shops, then the cashier must restart their system.
- **Pre-Condition**
  The user must have a points in his/he wallet to be redeemed.
- **Post-Condition**
  The user is able to redeem their rewards from the partnered shop by making a purchase from the shop.

8. View Integrated Shops

- **Main Success Scenario**

Actor Action	**System Response**
1. The user clicks on the button with the location icon	
	2. The system displays a list of partnered shops where the user can go to redeem their rewards.
	3. The system displays the exact location of the shop within the organization

- **Actor**
  Recyclist
- **Scope**
  Trash To Treasure System
- **Extensions**
  If the user is unable to view the list of partnered shops he must refresh the application and check his/her internet connection.
- **Pre-Condition**
  The user must be logged in to the system and connected to a stable internet connection.
- **Post-Condition**
  The user is able to view a list of all the partnered shop with the system.

9. Give User Feedback

- **Main Success Scenario**

Actor Action	System Response
1. The user clicks the reviews button from the side bar	
	2. the system will display all the reviews given by different users of the mobile application
	3. The system will ask the user to give their feedback if they wish to do so.
1. The user adds a user feedback.	
	3. The system displays the user's feedback in the application's reviews list.

- **Actor**
  Recyclist
- **Scope**
  Trash To Treasure System
- **Alternate Flow**
  3a. If the users do not wish to give their feedback they can just view other's reviews.
- **Extensions**
  If the user is unable to view the user reviews he must refresh the application and check his internet connection.
- **Pre-Condition**
  The user must be logged in to the system and connected to a stable internet connection.
- **Post-Condition**
  The user is able to view a list of all the user reviews given to the system (unless removed by the admin)

10. Register on Web Application

- **Main Success Scenario**

Actor Action	System Response
1. Cashier clicks the register button.	
	2. System asks the user to create a new account or login with an existing account.
3. Cashier chooses to create a new account option	
	4. System asks the user to enter his/her shop's full name, their full name, email address, mobile number, and a new password for the account
5. Cashier enters his/her details then clicks next button	
	6. System sends an OTP to the user's mobile phone and asks to enter the OTP for verification of the account created.
7. Cashier enters the OTP	
	8. System verifies the user and shows a message saying account created successfully

- **Actor**
  Cashier
- **Scope**
  Trash To Treasure System
- **Alternate Flows**
  2a. If the account is already created user chooses the login option
  2b. System asks the user to enter the email address and password.
  2c. User enters the email address and password and is logged in to the system.
- **Extensions**
  If at any time the system fails, user restarts the web application
- **Pre-Condition**
  The user must have a desktop/laptop setup to operate the web application freely.
- **Post-Condition**
  The cashier registration is successful.

11. Scan QR Code (for web app)

- **Main Success Scenario**

Actor Action	System Response
1. Cashier opens the scan QR code feature of theri web application.	
	2. System displays a screen with camera open to scan the QR code provided by the mobile application user.
3. Cashier scans the QR code.	
	4. The system displays the information stored in the QR code on the web application

- **Actor**
  Cashier
- **Scope**
  Trash To Treasure System
- **Extensions**
  a. If at any time the QR code scanning process generates an error the cashier tries again
  b. If at anytime the screen goes down user enters pin code instead.
- **Pre-Condition**
  The cashier must be logged in to the system and have stable internet connection.
- **Post-Condition**
  a. The QR code is scanned successfully and its information is displayed to the cashier.
  b. The status of the QR code and its associated pin code is updated to redeemed and its validity expires.

12. Enter Pin Code (for web app)

- **Main Success Scenario**

Actor Action	System Response
1. Cashier clicks the 'Enter Pin Code option from the side.	
	2. System displays a screen which asks the cashier to enter the pin code displayed on the trash bin screen.
3. Cashier Enters that pin code manually into the web app which is dictated by the customer.	
	4. The system displays the information stored in the pin code on the web application

- **Actor**
  Recyclist
- **Scope**
  Trash To Treasure System
- **Extensions**
  a. If the screen says invalid pin code user tries to enter the pin code again.
  b. If anytime the screen goes down the user goes for the Scanning QR code option instead.
- **Pre-Condition**
  The user must be logged in to the system.
- **Post-Condition**
  a. The pin code is successfully entered in the system and user information is displayed.
  b. The status of the pin code and its associated QR code is updated to redeemed and its validity expires.

13. Track User Points

- **Main Success Scenario**

Actor Action	System Response
1. The cashier can view the existing user points in their wallet after scanning the QR code provided.	
	2. The system then calculates the discount that is to be provided to the user on their purchase, according to the defined point system.
3. The cashier tells the discount provided to the buyer and confirms their purchase in their system	
	4. The system deducts the used points from the user's wallet and updates the remaining points accordingly.
5. The cashier views the remaining points on their system and verbally confirms with the buyer.	

- **Actor**
  Cashier
- **Scope**
  Trash To Treasure System
- **Alternate Flow**
  1a. If the use does not have a wallet and presents a code combination to the cashier asking for discounts, then the cashier will enter that code into their system.
  2b. The system then calculates the discounts according to the points associated with that specific code combination.
- **Extensions**
  If the cashier is unable to view the calculated discounts, they must wait or restart the system.
- **Pre-Condition**
  The cashier must have the QR code or the unique code combination for which they can calculate the discounts to be provided.
- **Post-Condition**
  The system shall be able to calculate discounts accurately and track the user's wallet points to deduct the used points accordingly.

14. Manage User Accounts

- **Main Success Scenario**

Actor Action	System Response
1. The admin views all the existing user accounts on the system and can search for any specific user account	
	2. The system displays all the existing user accounts to the admin on the admin web application.
3. The user chooses to create a account for the user themselves.	
	4. The system allows the admin to create an account System and asks the admin to enter new user's full name, email address, mobile number, and a new password for the account.
5. The admin enters all the required details and confirm account creation.	

- **Actor**
  Admin
- **Scope**
  Trash To Treasure System
- **Alternate Flow**
  4a. If the admin chooses to delete the account they can do it by clicking the delete button.
  4b. If the admin update the account details, they can do so by clicking the update button.
- **Extensions**
  If the admin is unable to view all the existing users, they must wait or restart the system.
- **Pre-Condition**
  They must be logged in to the system and have stable internet connection.
- **Post-Condition**
  The admin shall be able to view and modify the information of all the existing users.

15. Manage Integrated Shops

- **Main Success Scenario**

Actor Action	System Response
1. The admin views all the existing integrated shops accounts on the system and can search for any specific account	
	2. The system displays all the existing shops integrated, to the admin on the admin web application.
3. The user chooses to create a account for the cashier themselves.	
	4. The system allows the admin to create an account and asks the admin to enter the shop's full name, their full name, email address, mobile number, and a new password for the account
5. The admin enters all the required details and confirm account creation.	

- **Actor**
  Admin
- **Scope**
  Trash To Treasure System
- **Alternate Flow**
  4a. If the admin chooses to delete the account they can do it by clicking the delete button.
  4b. If the admin update the account details, they can do so by clicking the update button.
- **Extensions**
  If the admin is unable to view all the partnered shops, they must wait or restart the system.
- **Pre-Condition**
  They must be logged in to the system and have stable internet connection.
- **Post-Condition**
  The admin shall be able to view and modify the information of all the partnered shops.

16. Manage Point System

- **Main Success Scenario**

Actor Action	System Response
1. The admin views the point system defined	
2. The admin click the modify button to modify the existing point system.	
	3. The system allows the admin to modify or change the values of the existing point system
	4. The system asks the admin to enter the values that tells the points that will be given to the user based on the type of trash which includes, plastic, paper or metal.
5. The admin enters different values and click the confirm button to modify the values.	

- **Actor**
  Admin
- **Scope**
  Trash To Treasure System
- **Extensions**
  If the admin is unable to modify the point system values, then they must check their internet connection and try again.
- **Pre-Condition**
  The admin must be logged into the system with a stable internet connection.
- **Post-Condition**
  The admin shall be able to modify the point system successfully.

17. Manage User Feedback

- **Main Success Scenario**

Actor Action	System Response
1. The admin views the feedback given by all the users of the system.	
2. The admin clicks delete button to delete a specific user feedback.	
	3. The system deletes the user feedback from the system.

- **Alternate Flow**
  1a. If the admin wishes they can reply to specific user feedbacks. 1b. The admin's reply will be visible to everyone on the mobile application.
- **Extensions**
  If the admin is unable to reply, they must check their internet connection and try again.
- **Pre-Condition**
  The admin must be logged in to the system with a stable internet connection
- **Post-Condition**
  The admin shall be able to view all the user feedbacks, delete them or reply to them as required.

2.2 Storyboarding

2.2.1 Storyboard of Hardware Bin

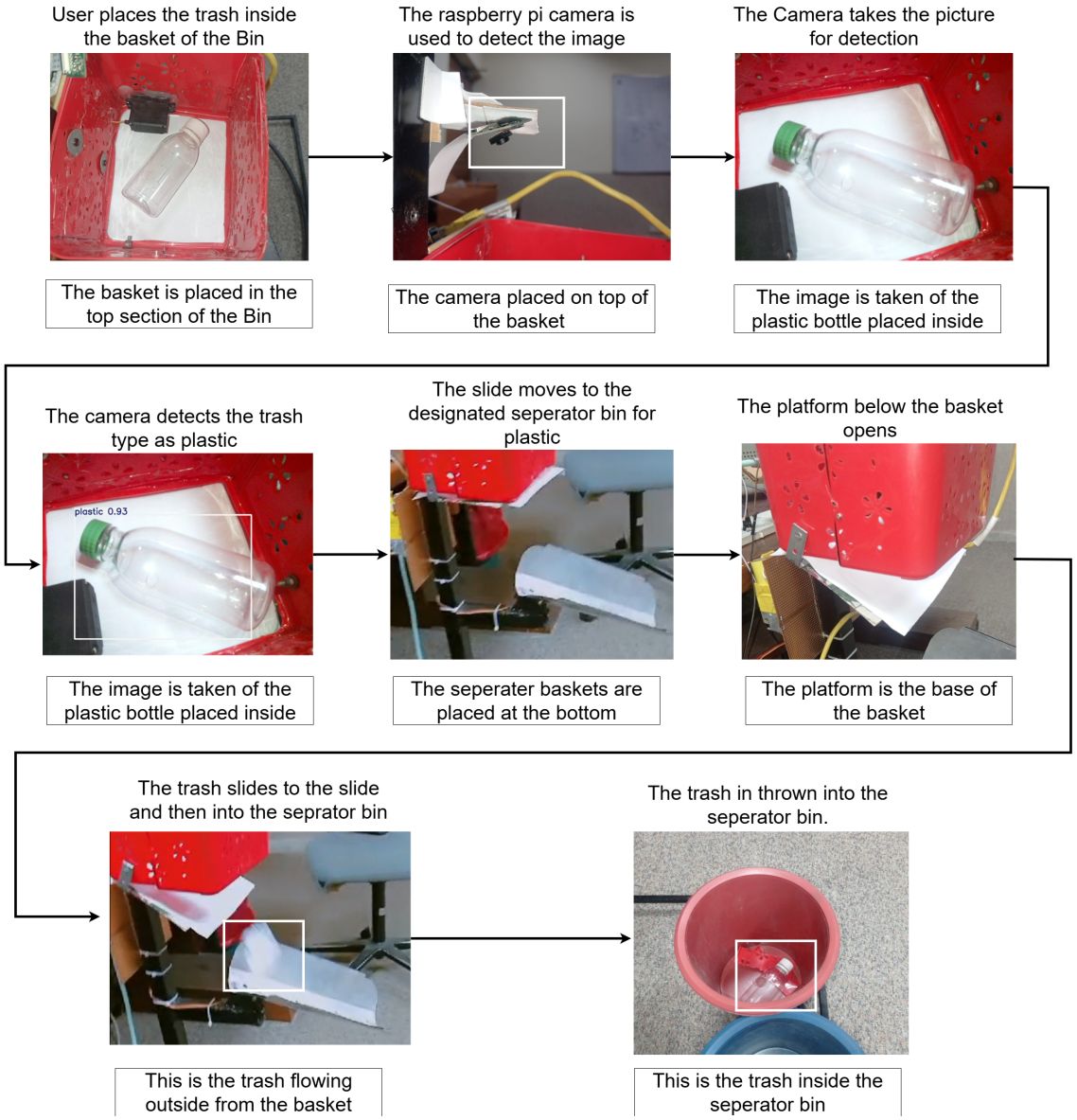


Figure 2.2: Storyboard of Hardware Bin

2.3 Hardware Structure

Figure 2.3: Hardware Structure

2.3.1 Storyboard of Client Mobile App - Part 1

Figure 2.4: Storyboard of Client Mobile App - Part 1

2.3.2 Storyboard of Client Mobile App - Part 2

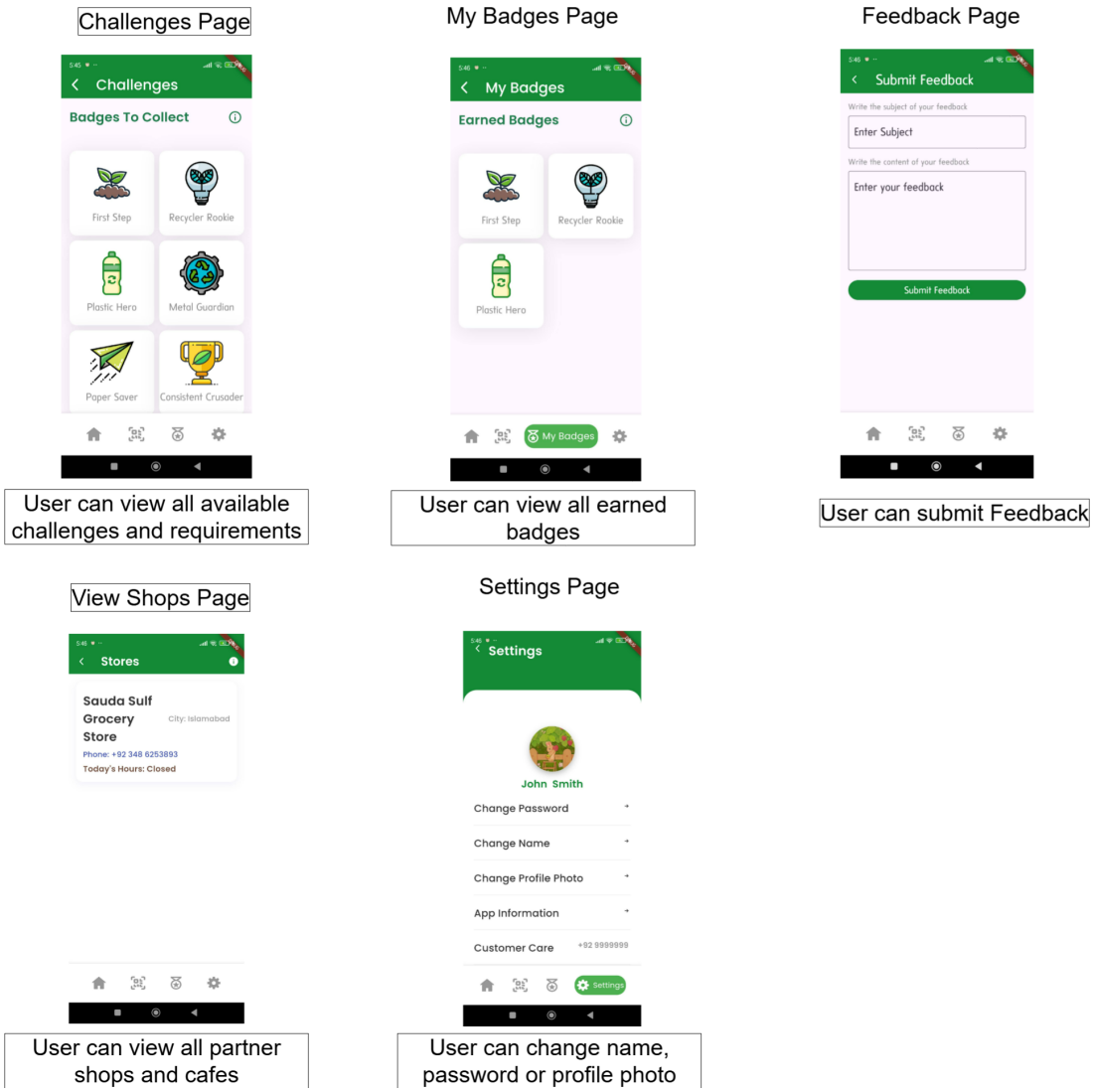
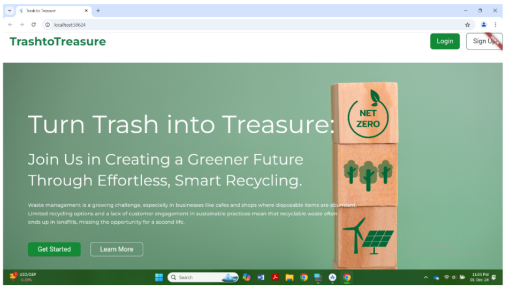


Figure 2.5: Storyboard of Client Mobile App - Part 2

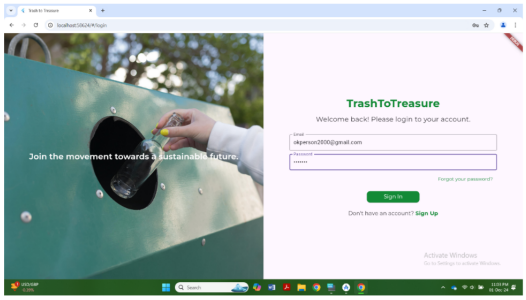
2.3.3 Storyboard of Client Web App - Part 1

Main Dashboard



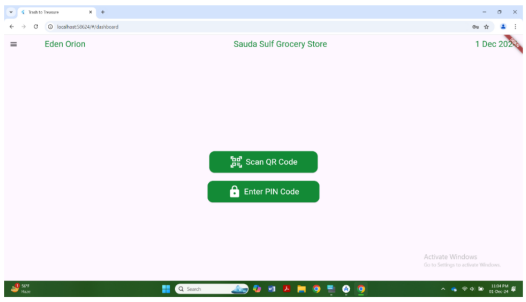
New Users can sign up and get an introduction to our product. Existing Users can sign in.

Sign In Page



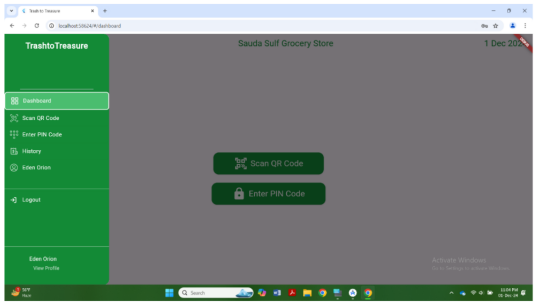
User enters details to sign in

Main Dashboard of Cashier



User can scan QR, enter PIN or change settings

Main Dashboard of Cashier with drawer



User can view different features

Figure 2.6: Storyboard of Client Web App - Part 1

2.3.4 Storyboard of Client Web App - Part 2

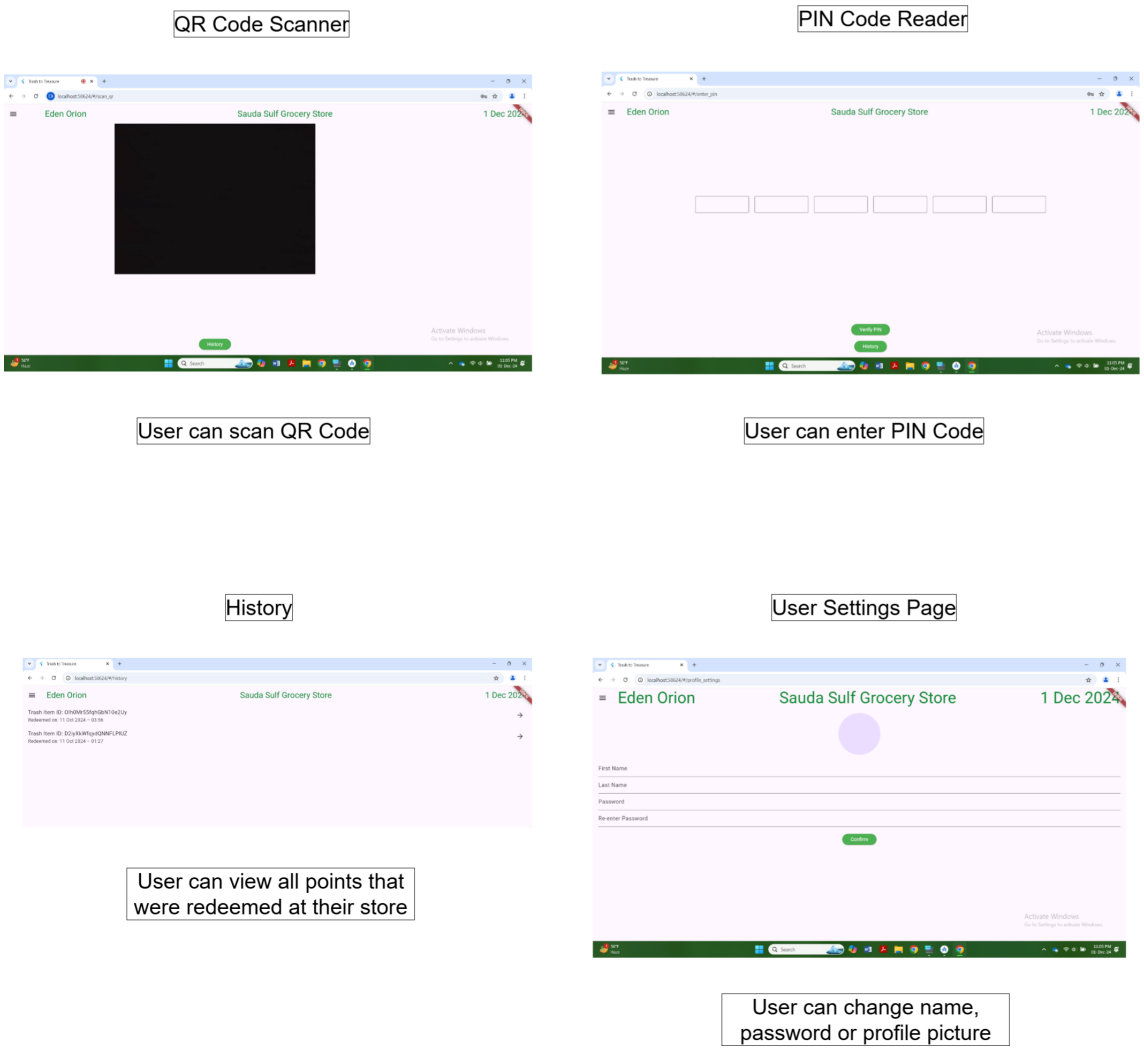


Figure 2.7: Storyboard of Client Web App - Part 2

2.4 Functional Requirements

2.4.1 Data Collection and Pre-processing Module

2.4.1.1 Data Collection from Kaggle and University Cafeteria

1. The system shall integrate with Kaggle datasets to acquire relevant image data for training.
2. The system shall provide functionality to collect additional image samples from the university cafeteria for training purposes.

2.4.1.2 Pre-processing of Image Data

1. The system shall perform pre-processing of the collected image data to prepare it for training.
2. The system shall normalize pixel values within the images to a specific range to ensure consistency.
3. Augmentation techniques such as rotation, flipping, and cropping shall be applied to increase dataset diversity.

2.4.1.3 Feature Extraction

1. The system shall extract relevant features from pre-processed images that are essential for the trash classification task.
2. Feature extraction techniques shall be employed to identify distinctive characteristics of different waste materials.

2.4.1.4 Training of Object Classification Model

1. The system shall facilitate the training of an object classification model using the pre-processed dataset.
2. Training shall involve optimizing the model parameters to improve classification accuracy.
3. The system shall employ machine learning algorithms or deep learning frameworks for model training.

2.4.2 AI-based Trash Classfication Module

2.4.2.1 Object Detection and Classification

1. The system shall utilize the trained object detection model to identify and classify different types of trash.
2. Trash items shall be categorized into predefined categories such as plastic, metal, and paper based on their visual features.

2.4.2.2 Identification of Mixed Waste

1. If the detected object does not belong to any of the predefined categories (plastic, metal, paper), the system shall classify it as mixed waste.
2. No points or rewards shall be awarded for items classified as mixed waste.

2.4.2.3 Integration with Trash Collection Process

1. The AI-based Trash Classification Module shall seamlessly integrate with the trash collection process facilitated by the Smart Bin module
2. Trash items deposited into the Smart Bin shall be automatically analyzed and classified by the AI model before further processing.

2.4.2.4 Real-time Processing

1. The system shall perform trash classification in real-time to provide immediate response to users upon disposal of items.
2. Real-time processing capabilities shall ensure efficient and timely recognition of waste materials.

2.4.3 Smart Bin Module

2.4.3.1 Trash Disposal Mechanism by Recyclist

1. The Smart Bin shall feature a platform onto which trash items are deposited upon disposal by the user.
2. Once a trash item is thrown, it shall slide onto the platform, preventing retrieval by the user.

2.4.3.2 Integration with AI-based Trash Classification Module

1. The Smart Bin shall be equipped with a camera positioned above the platform to capture images of disposed trash items.
2. Captured images shall be transmitted to the AI-based Trash Classification Module for classification and categorization.

2.4.3.3 Automated Trash Sorting

1. Upon classification of a trash item by the AI model, the main slide of the Smart Bin shall rotate to align with the opening of the corresponding smaller bin designated for the identified category.
2. Rotation of the main slide shall be controlled by a stepper motor integrated into the Smart Bin.

2.4.3.4 Trash Disposal Mechanism by Smart Bin

1. Once the main slide is aligned with the appropriate smaller bin, a small door on the platform shall open with the assistance of a servo motor.
2. The classified trash item shall then descend into the designated smaller bin through the open door.

2.4.3.5 QR Code Generation

1. Upon classification of a trash item, the Smart Bin shall generate a QR code containing relevant information such as the category of the trash and additional tracking details.
2. The generated QR code shall be displayed on a screen attached to the Smart Bin for user access.
3. Users not using the mobile application may manually note down the QR code displayed on the screen for later use.

2.4.3.6 Data Storage and Tracking

1. Details of the classified trash item along with the generated QR code shall be stored in the database integrated with the system.

2. Stored data shall be utilized for tracking user recycling behavior and awarding rewards based on points earned.

2.4.4 Client Mobile Application Module

1. The mobile application shall provide functionality for users to create and manage their profiles.

2. Users shall be able to set up profiles, track their recycling history, and customize their app preferences.

3. The application shall feature QR code scanning functionality, allowing users to scan QR codes associated with different types of trash.

4. Real-time point tracking shall be implemented to display users' current points earned through recycling activities.

5. Users must have a minimum of 1000 points available to be eligible for redemption.

6. The application shall maintain a history of users' recycling activities, including dates of recycling activities, types of materials recycled, points earned, and rewards redeemed.

7. Gamification elements such as challenges, badges, or a point-based leveling system shall be introduced to enhance user engagement and promote consistent recycling behavior.

8. Users shall have the ability to send feedback to the administration of the Trash to Treasure system through the application.

9. The application shall provide users with access to detailed information about partnered stores and cafes.

10. Users shall be able to identify locations where they can redeem their rewards.

2.4.5 Client Web Application Module

1. The web application shall provide functionality for shops and cafes to easily register on the platform

2. Registration process shall be user-friendly and intuitive, allowing businesses to onboard efficiently.

3. QR code scanning functionality shall be implemented to enable cashiers at shops and cafes to scan QR codes provided by customers.

4. Scanning QR codes will ensure that points have not been redeemed already and/or expired.

5. The web interface shall be integrated with the point system for seamless transactions.

6. Cashiers shall be able to update the status of redeemed points in the database only, ensuring accurate tracking of point redemption.

2.4.6 Admin Web Application Module

1. The admin web application shall provide an interface for admins to manage user accounts and access.

2. Admins shall have the ability to create, modify, and deactivate user accounts as needed.

3. The admin web application shall allow admins to manage the overall point system.

4. Admins shall be able to adjust point values, set rewards, and monitor point distribution to users.

5. The admin web application shall feature a console for admins to manage registered shops and cafes.

6. Admins shall be responsible for approving registrations and ensuring compliance with the platform's standards for partnered businesses.

7. Admins shall have the ability to review and respond to user feedback, as well as track trends and issues reported by users.

2.5 Non-Functional Requirements

2.5.1 Reliability

1. The system as a whole shall have a MTBF of at least 150 hours.

2. Failure is defined as any event that results in system downtime, data loss, or incorrect functionality.

3. In the event of system failure, users may experience delays or inability to access system functionalities.

4. Failure to accurately classify trash items may result in misallocation of resources for waste management.

5. Automatic error recovery mechanisms shall be in place to mitigate the impact of failures and restore normal operation.

2.5.2 Usability

1. The system shall feature user-friendly interfaces across all modules, ensuring ease of learning and use.

2. Users shall be able to complete tasks such as recycling, point tracking, and redemption without encountering usability issues.

3. The system shall respond to user inputs within 3 seconds for all primary functions, ensuring efficient interaction.

4. Users shall have the ability to undo actions or navigate back to the previous screen to correct errors.

2.5.3 Performance

1. The system shall have a maximum response time of 3 seconds for all user interactions, including trash classification, point tracking, and redemption.

2. The system shall be capable of handling concurrent requests from 1000 users without degradation in performance.

2.5.4 Security

1. The system shall encrypt sensitive user data, such as personal information and transaction history, using industry-standard encryption algorithms.

2. The system shall implement role-based access control to ensure that only authorized personnel can access administrative functions.

.

Chapter 3

System Overview

The project aims to develop a system for trash classification and rewards that encourages sustainable waste practices. It involves several modules: Data Collection and Preprocessing, AI-based Trash Classification Module, Smart Bin, Client Mobile App, Client Web App, Admin Web App.

3.1 Architectural Design

3.1.1 Box and Line Diagram

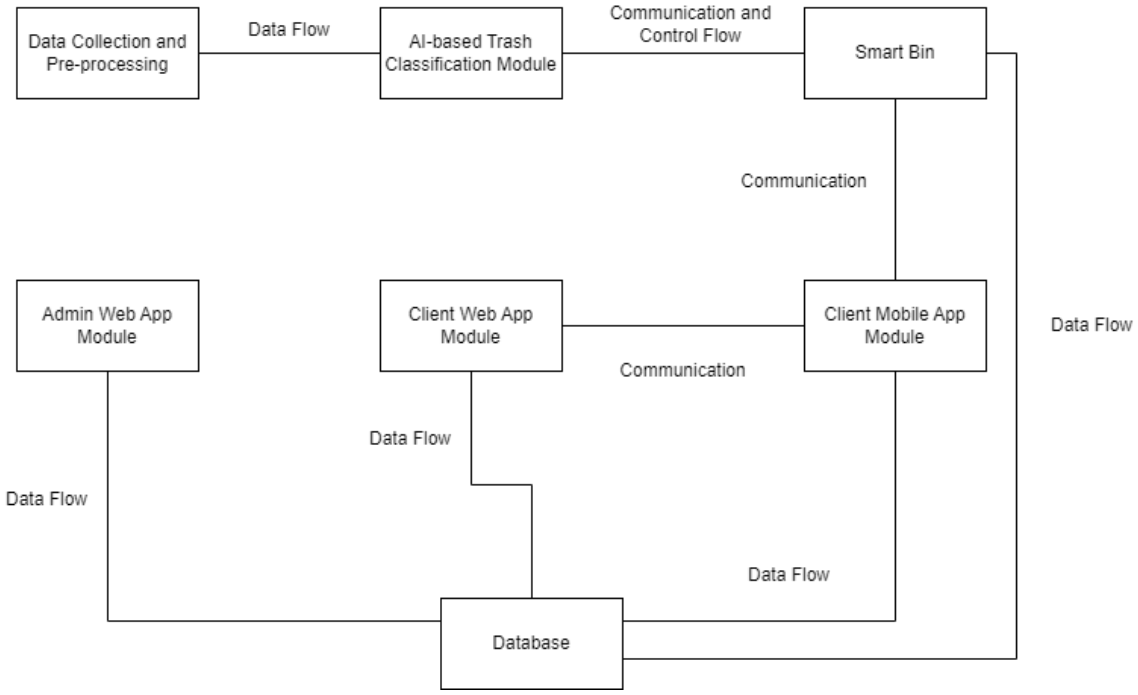


Figure 3.1: Box and Line Diagram

3.1.2 Layered Architecture Diagram

Presentation Layer: This layer represents the user interface components that interact directly with the users. In our project, the Client Mobile App, Admin Web App and Client Web App modules serve as the presentation layer.

Business Logic Layer (Application Layer): This layer contains the core functionality and business rules of the system. It processes and manipulates data according to the business requirements. In our project, the Data Collection and Pre-processing Module, AI-based Trash Classification Module, the Smart Bin resides in this layer.

Data Access Layer: Also known as the Persistence Layer, this layer is responsible for managing the storage and retrieval of data from the underlying database or data storage system. In our project, the Database will serve as the Data Access Layer.

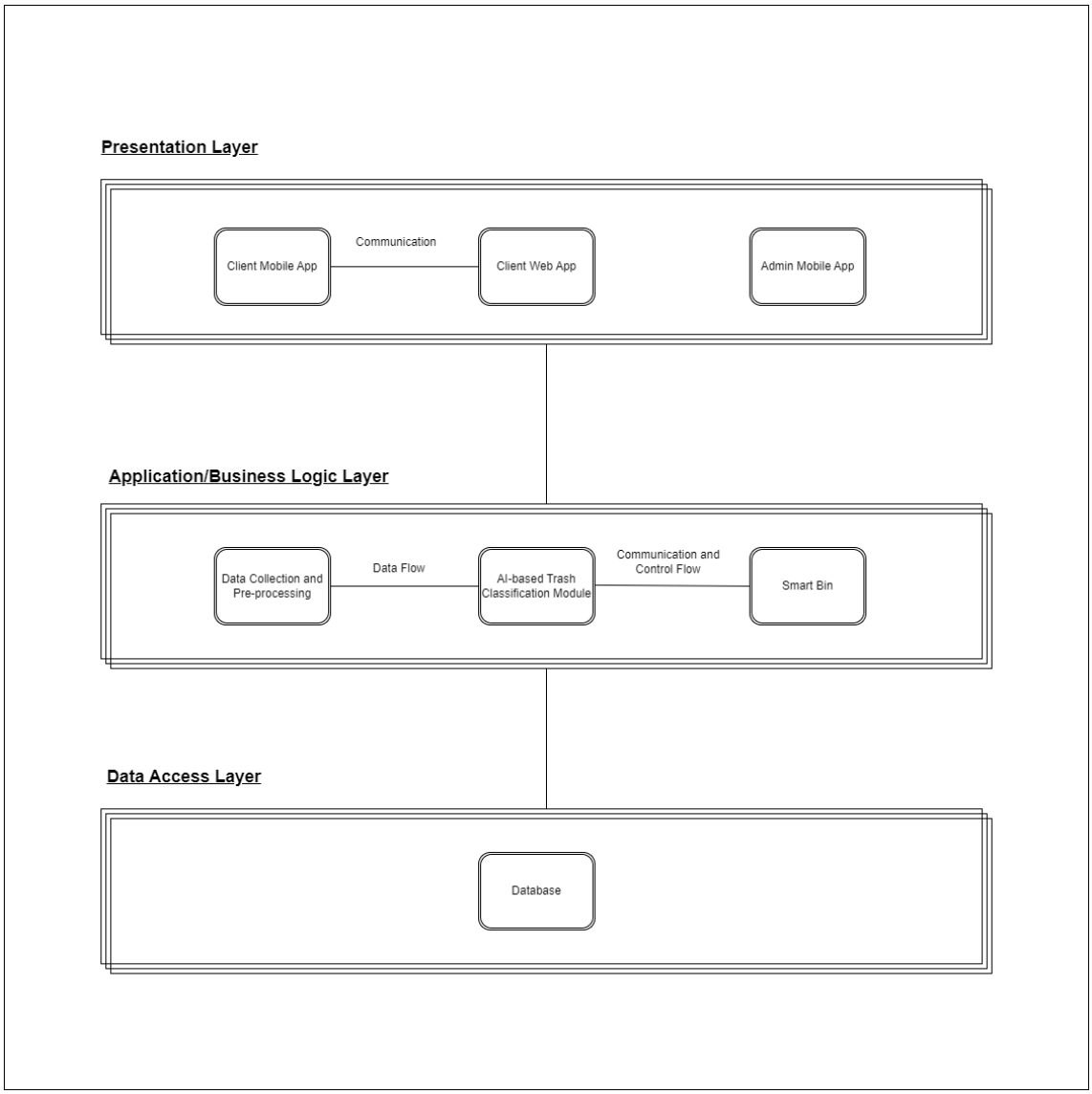


Figure 3.2: 3-Tier Layered Architecture Diagram

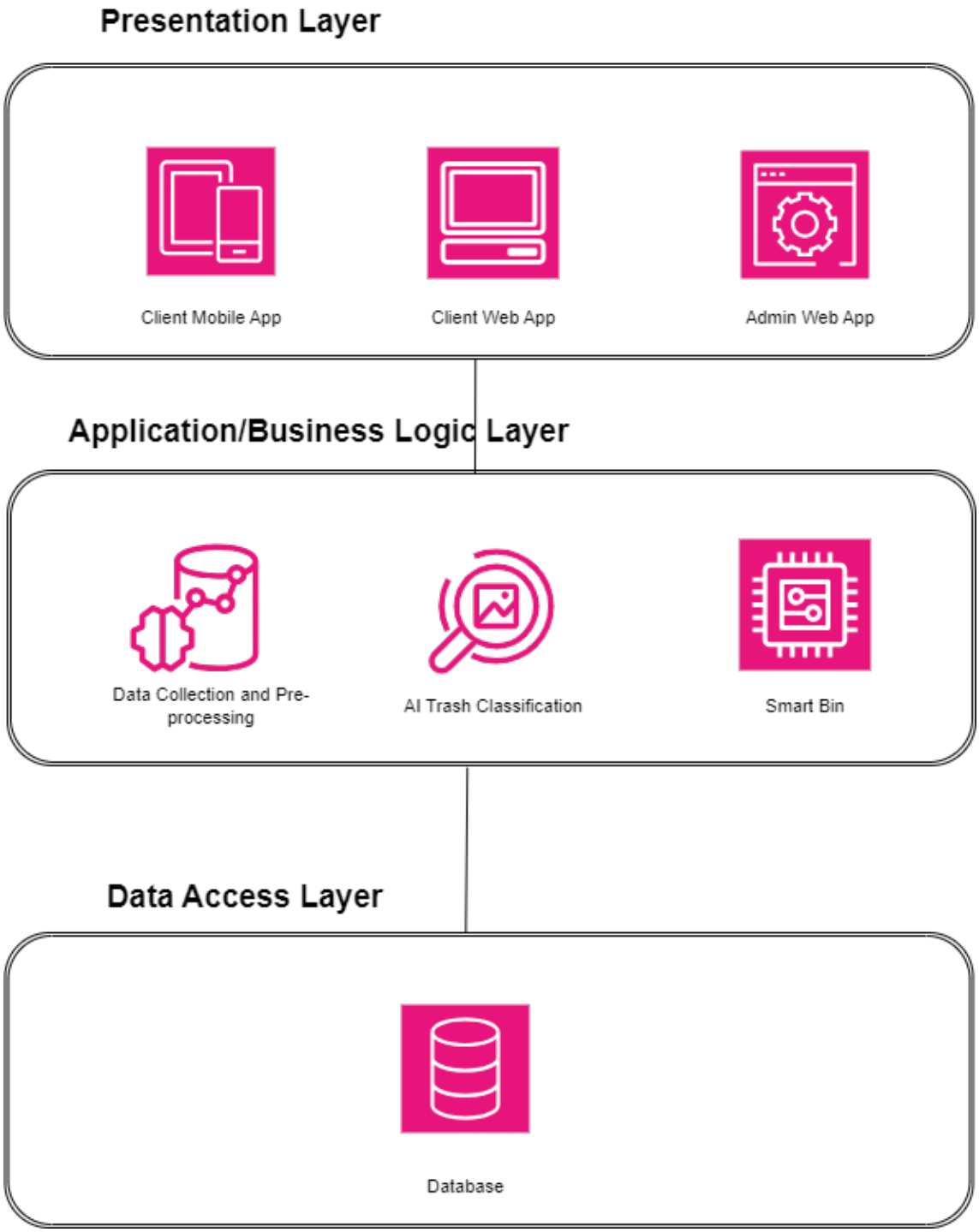


Figure 3.3: 3-Tier Layered Architecture Diagram

3.2 Design Models

3.2.1 Activity Diagram

3.2.1.1 Activity Diagram for Data Collection and Model Training

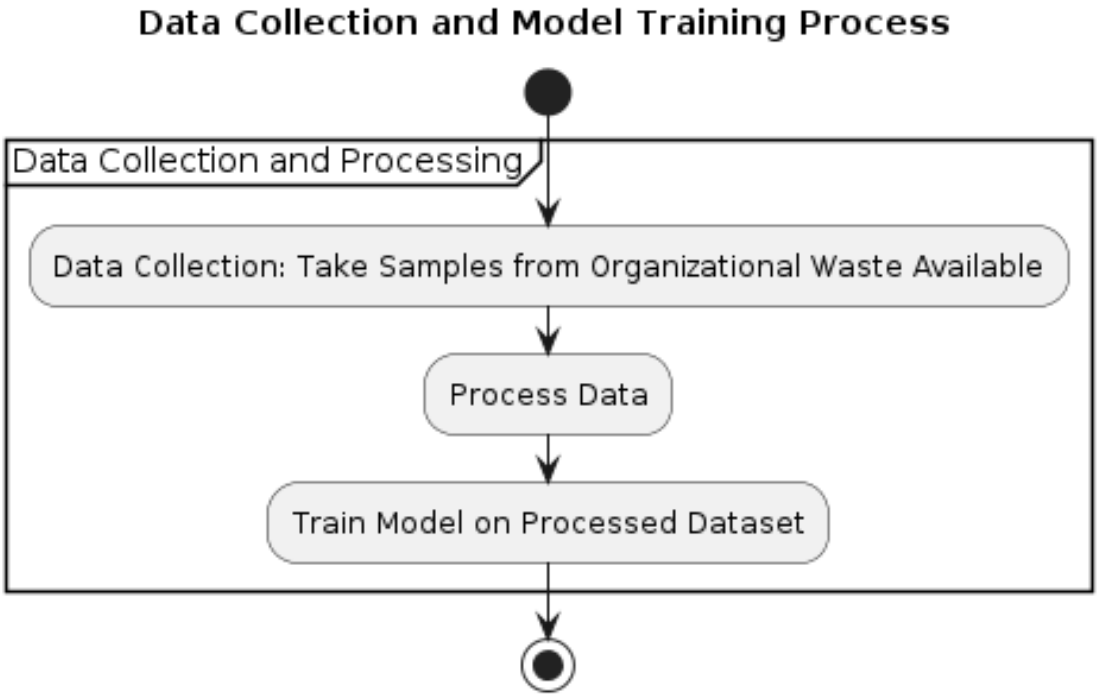


Figure 3.4: Data Collection and Model Training

3.2.1.2 Activity Diagrams for User Registration

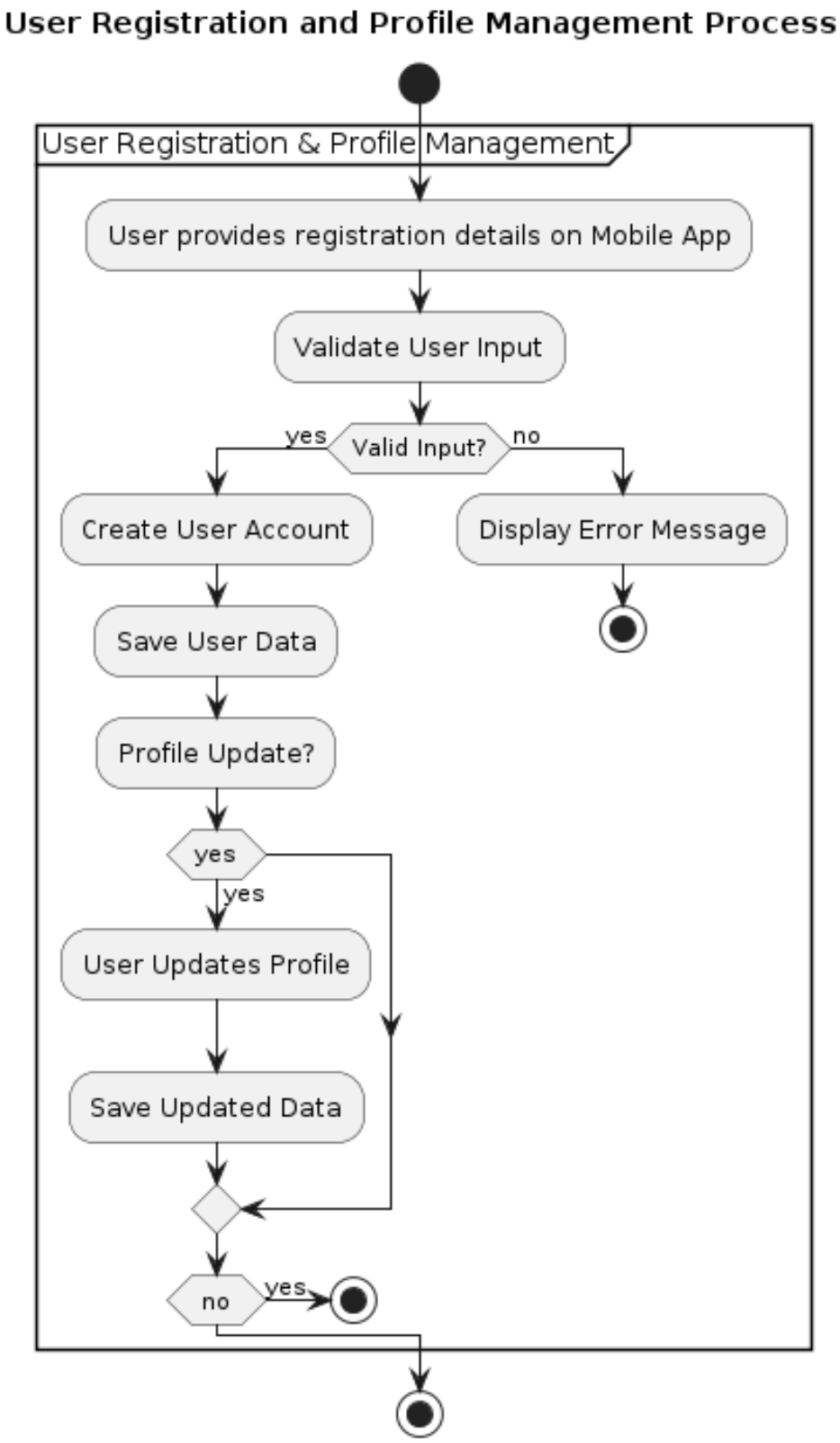


Figure 3.5: User Registration

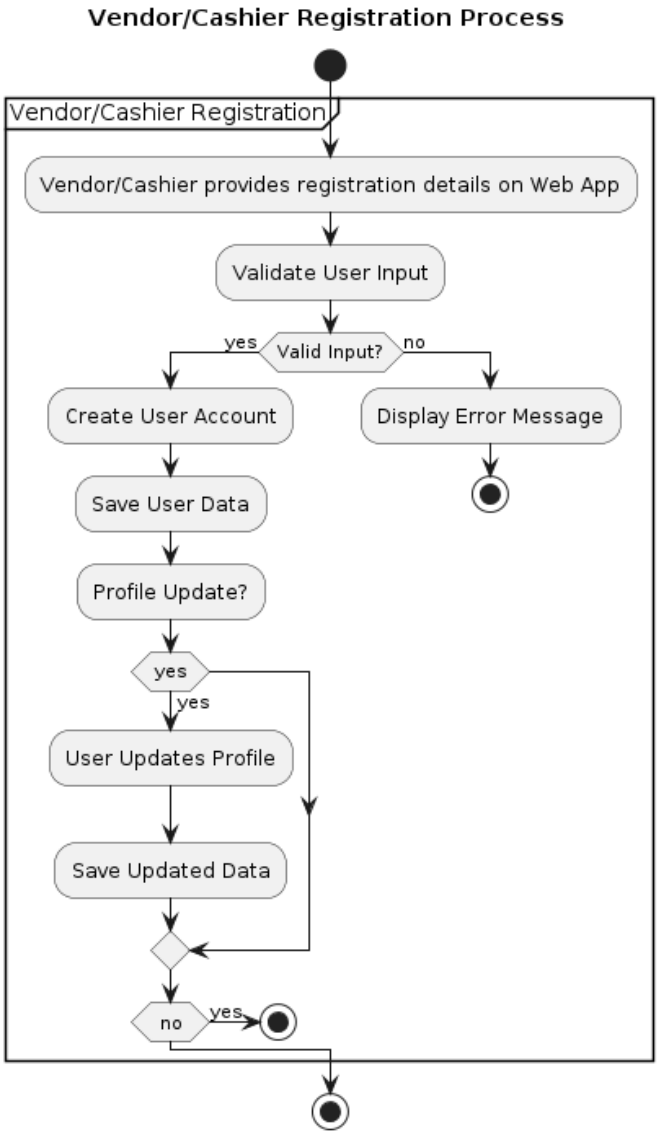


Figure 3.6: User Registration

3.2.1.3 Activity Diagram for Admin

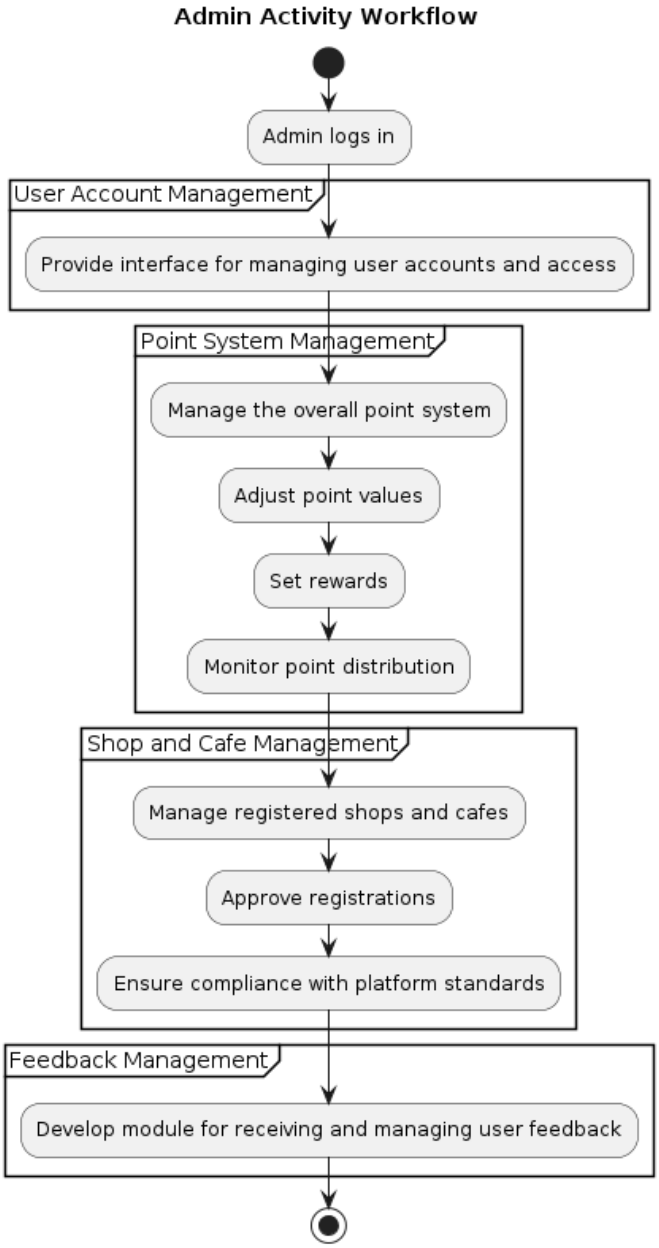


Figure 3.7: Activity Diagram for Admin

3.2.1.4 Activity Diagrams for Recyclist

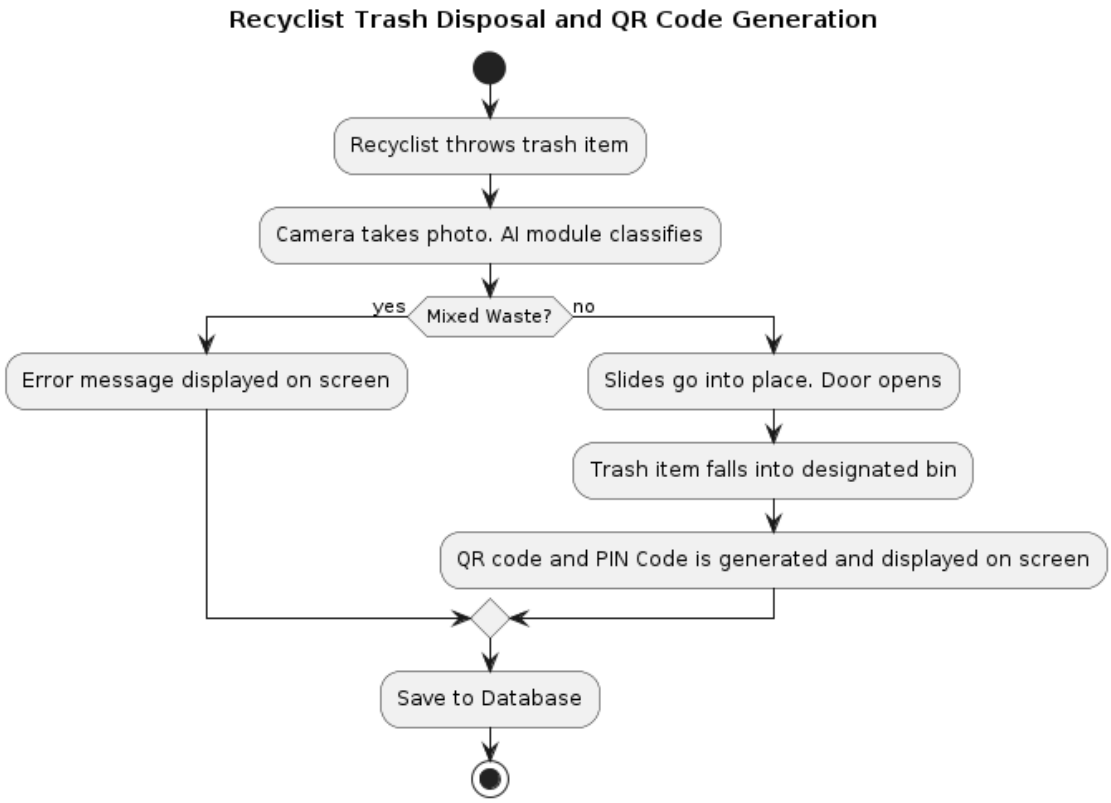


Figure 3.8: Activity Diagram for Trash Disposal and QR Code and PIN Code Generation

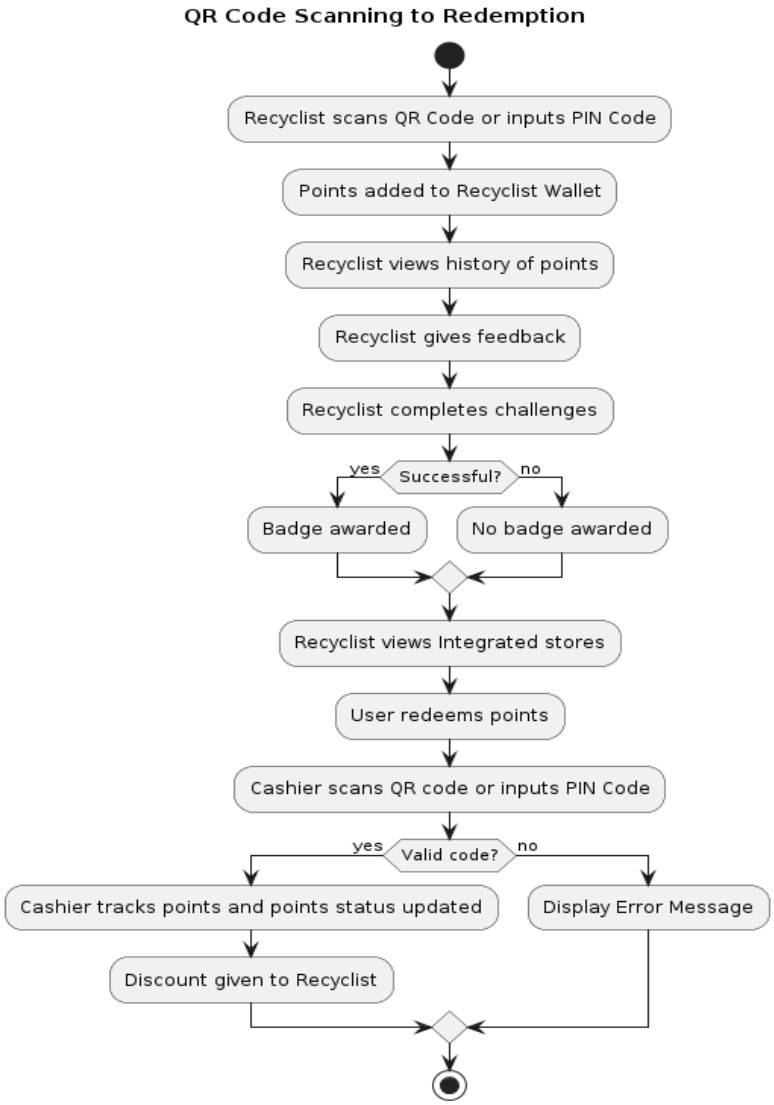


Figure 3.9: Activity Diagram for QR Code Scanning to Redemption

3.2.1.5 Activity Diagram for Cashier

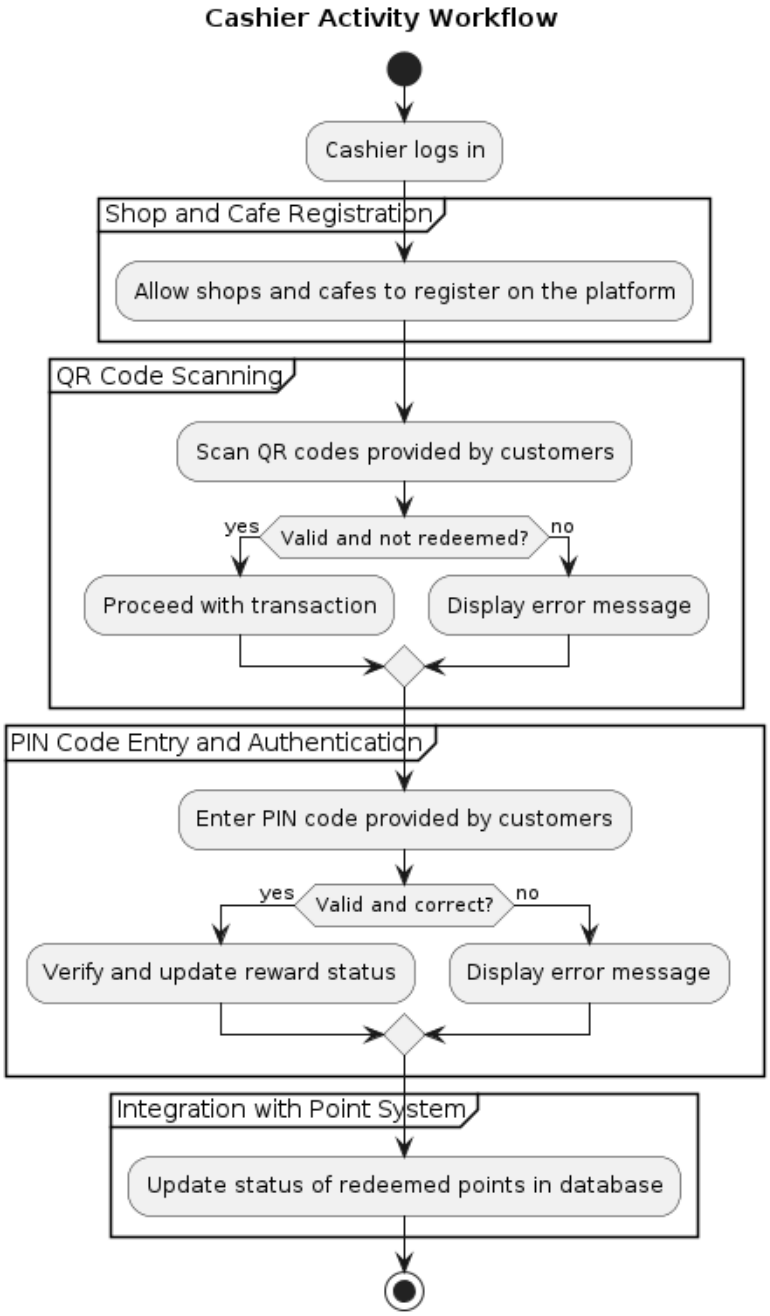


Figure 3.10: Activity Diagram for Cashier

3.2.2 Class Diagram

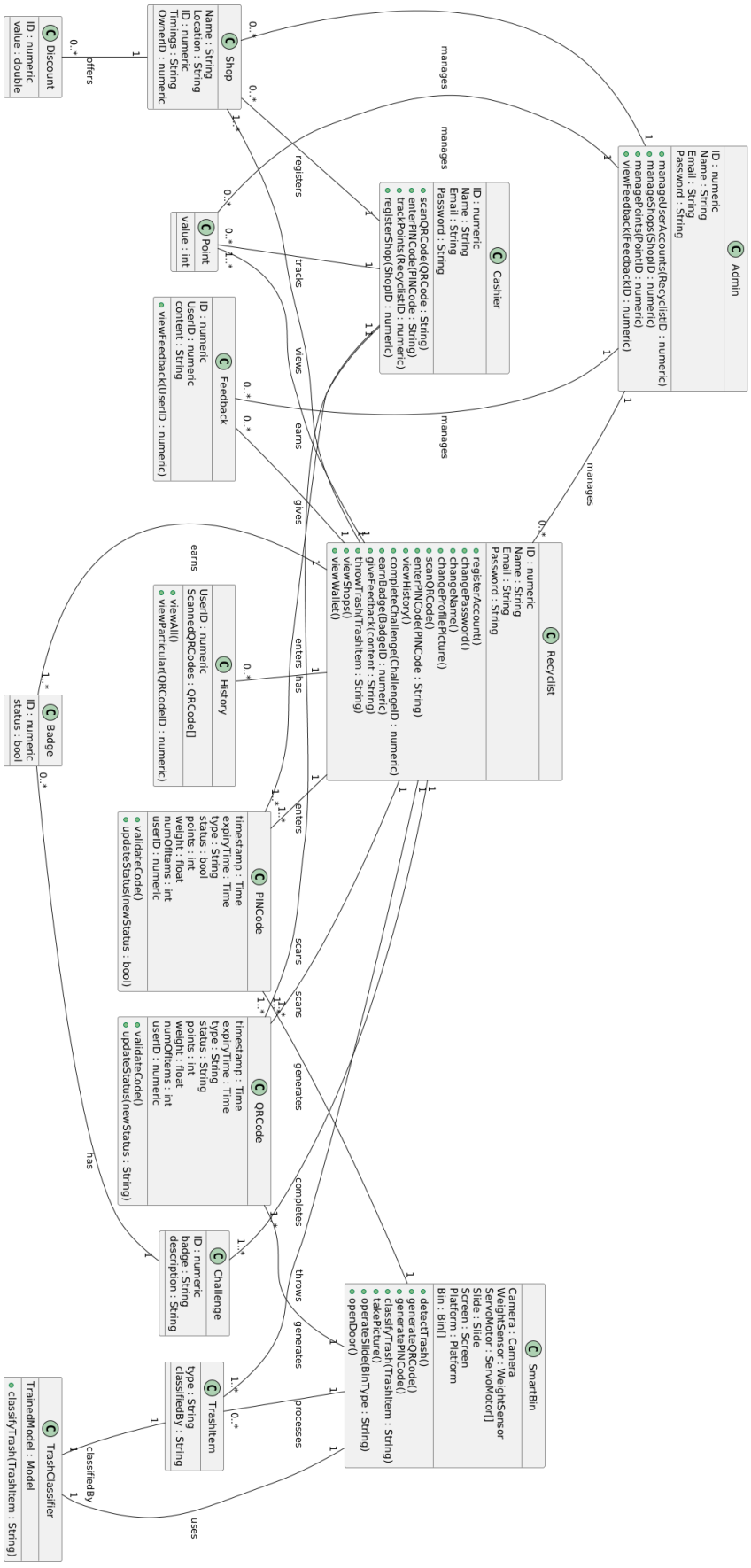


Figure 3.11: Detailed System Class Diagram

3.2.3 Class-Level Sequence Diagram

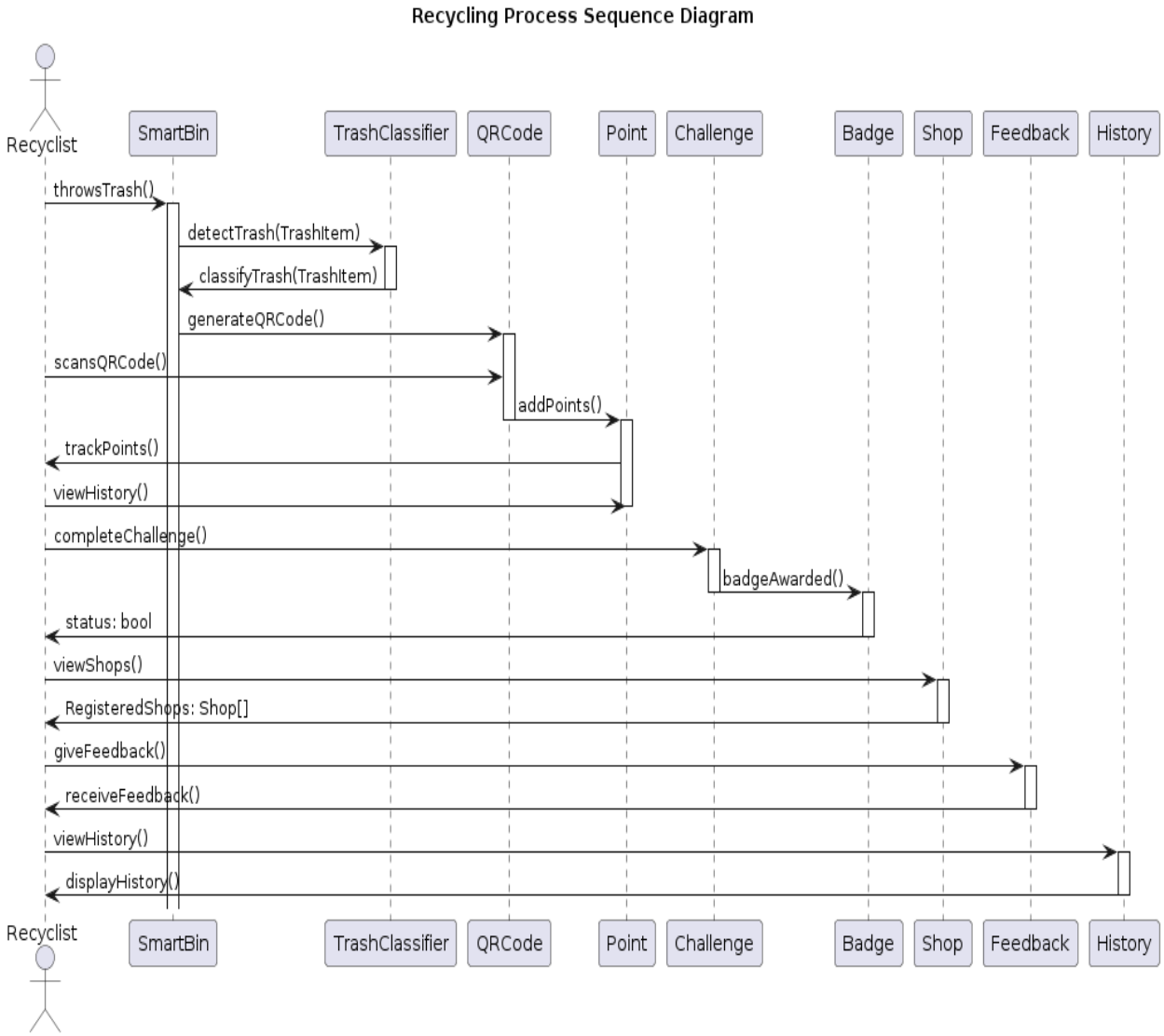


Figure 3.12: Class-Level Sequence Diagram for Recyclist

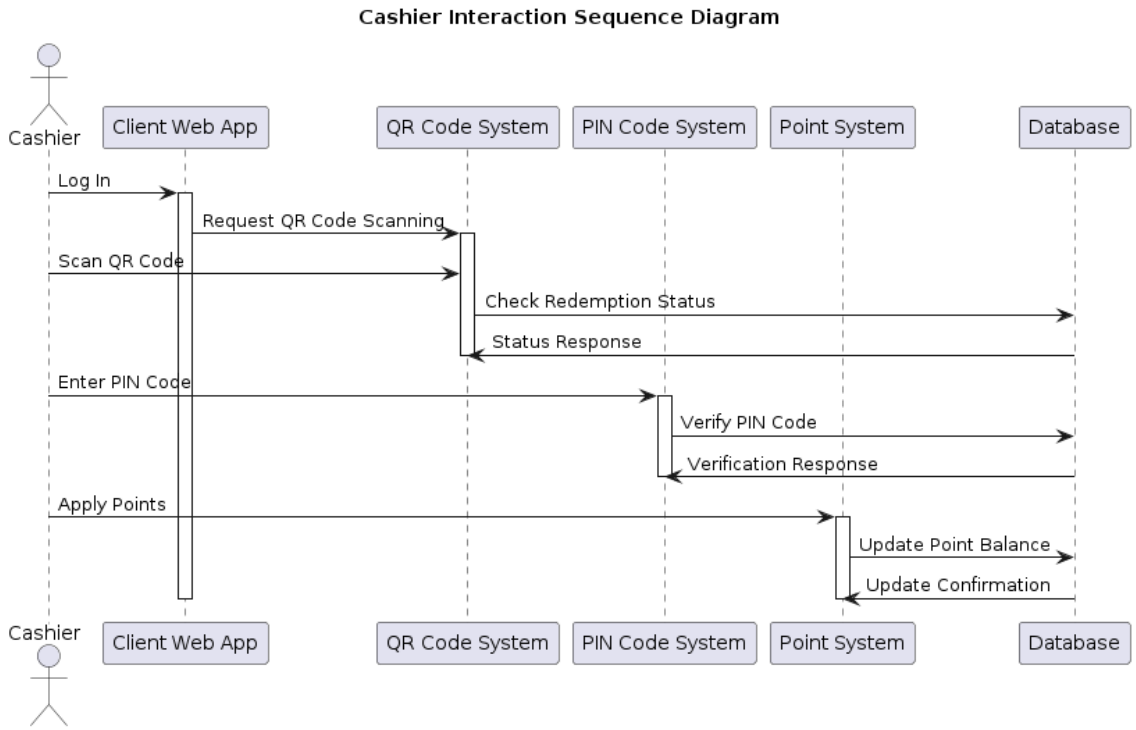


Figure 3.13: Class-Level Sequence Diagram for Cashier

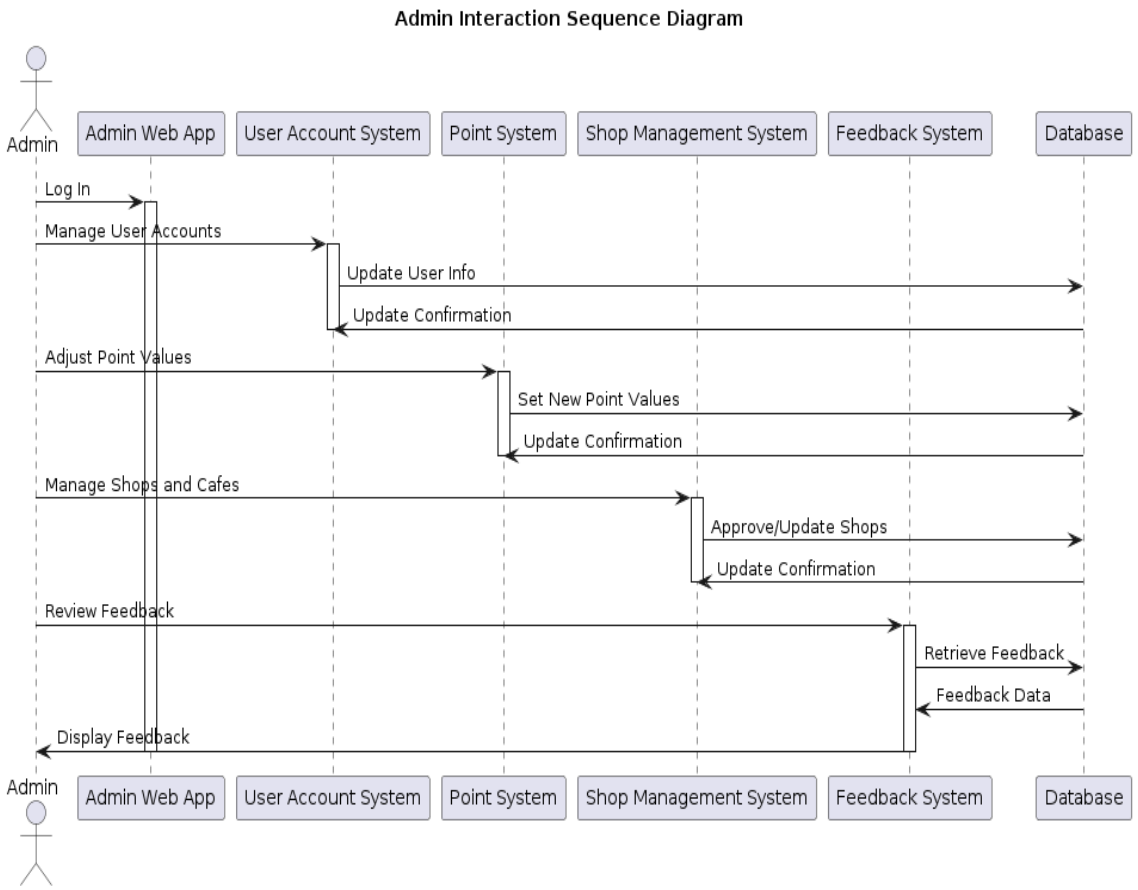


Figure 3.14: Class-Level Sequence Diagram for Admin

3.3 Data Design

1. **User profiles** - Contains user authentication data, preferences, and historical interaction data.

2. **Trash items** - Detailed records of each item of trash, including type, classification results, and associated QR codes.

3. **QR codes and PIN Codes** - Generated codes that link to specific trash item data and the associated reward points.

4. **Points earned by users** - Tracking of points which users accumulate through recycling activities.

5. **User Feedback** - Direct feedback from users about their experience, used for system improvements.

6. **Shop and cafe registrations** - Information about participating shops and cafes that accept points as part of a loyalty program.

7. **Admin account** - Admin profiles for managing the system, users, and configuration settings.

3.3.1 Data Transformations

1. **User registration data** is captured via the frontend and transformed into structured user profiles stored in the database. This includes setting up credentials and preferences.

2. **Shop and cafe registration forms** are processed to create detailed profiles for each shop, which are used to manage partnerships and redemption options for points.

3. **Trash items** are scanned by the system, classified, and each item is logged with a unique identifier (QR code and PIN Code) that links to its classification type and potential points value. The weight of each item is captured using sensors integrated with Raspberry Pi, providing precise data for analysis and points allocation.

4. **User feedback** is collected through the app interface and stored directly into the feedback management system for administrative review and action.

5. **Weight data from trash items** is continuously monitored and updated by sensors connected via Wi-Fi-enabled Raspberry Pi. This data is essential for calculating the volume and type of trash processed, optimizing recycling outcomes and operational efficiency.

3.3.2 Main Data Storage Items

1. **Databases**: We are utilizing Firebase Firestore, a NoSQL cloud database that provides flexible, real-time data synchronization and storage, suitable for our application's dynamic and interactive features.

2. **User Database**: Firestore stores all user profile information in collections, where each user profile is a document containing fields such as user credentials, preferences, and activity logs. This model supports flexible schema adjustments and seamless real-time updates.

3. **Trash Database**: Each trash item is represented as a document within a Firestore collection, which includes data on the type of trash, the results from the classification process, and references to the related QR codes and point allocations. Fire-

store's real-time capabilities ensure that this information is updated instantly across all client apps.

4. **Feedback Database**: User feedback is managed in a dedicated Firestore collection, where each piece of feedback is stored as a document. This allows administrators to easily query feedback data, respond to user concerns, and aggregate insights for service improvement.

5. **Shop and Cafe Database**: Firestore hosts documents for each partnered shop and cafe. Each document details the shop's profile, including point redemption rules and other relevant details. Firestore's scalability supports adding new shops and updating existing data without downtime.

6. **Challenges Database**: Challenges are tracked in their own Firestore collection, with documents detailing challenge specifics, user engagement, and reward mechanisms. Firestore's efficient data retrieval capabilities enable quick access to challenge statuses and updates, facilitating a responsive user experience.

7. **Sensor Data Database**: This database stores the weight data collected from sensors inside the Smart Bins. Each entry captures the weight measurements of trash items as they are deposited, enabling precise monitoring and analysis for optimization of waste management processes. This data is periodically updated to reflect new measurements, ensuring accurate and timely information for operational decisions.

3.3.3 Data Flow Diagrams

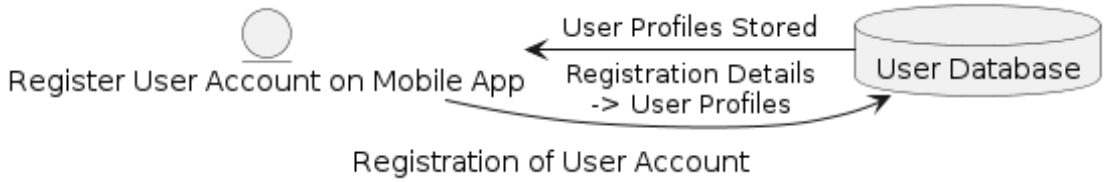


Figure 3.15: Data Flow Diagram for User Registration

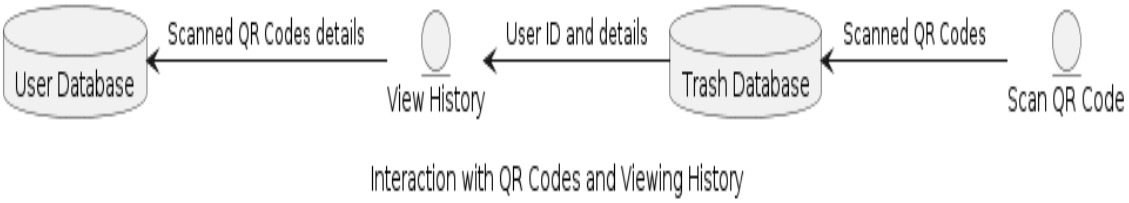


Figure 3.16: Data Flow Diagram for Scan QR Code and View History

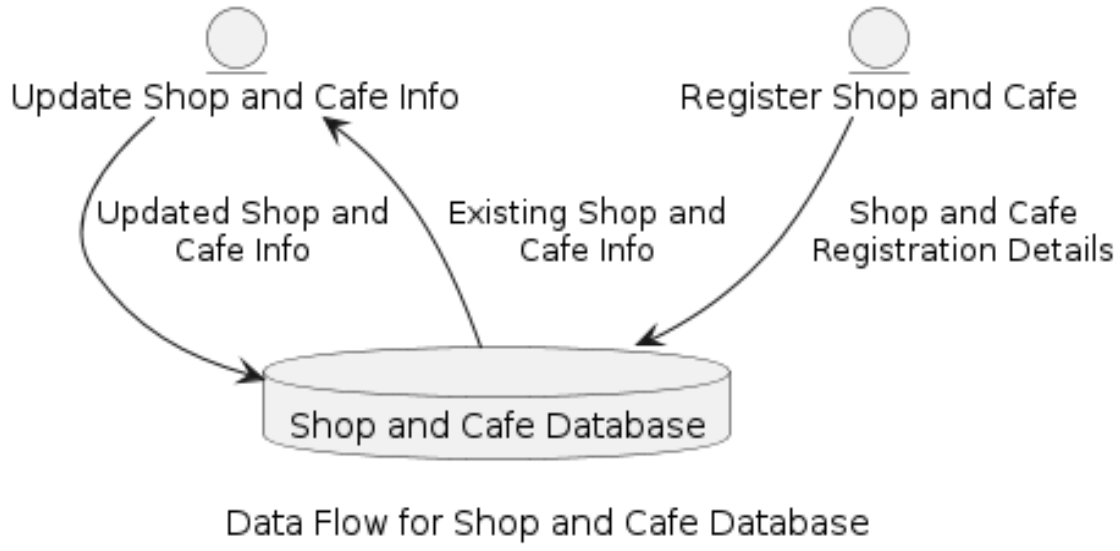


Figure 3.17: Data Flow Diagram for Shops

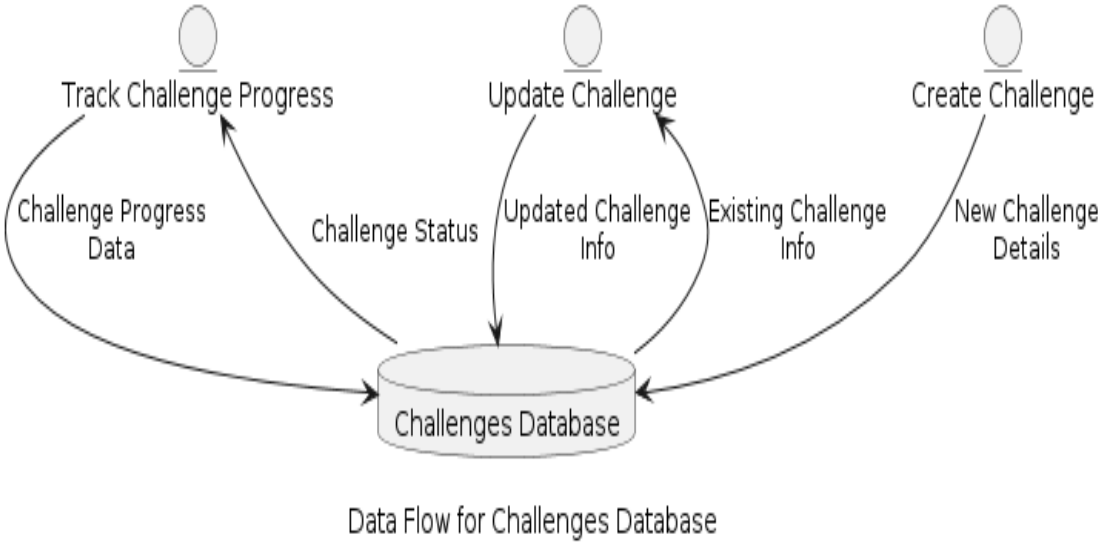


Figure 3.18: Data Flow Diagram for Challenges

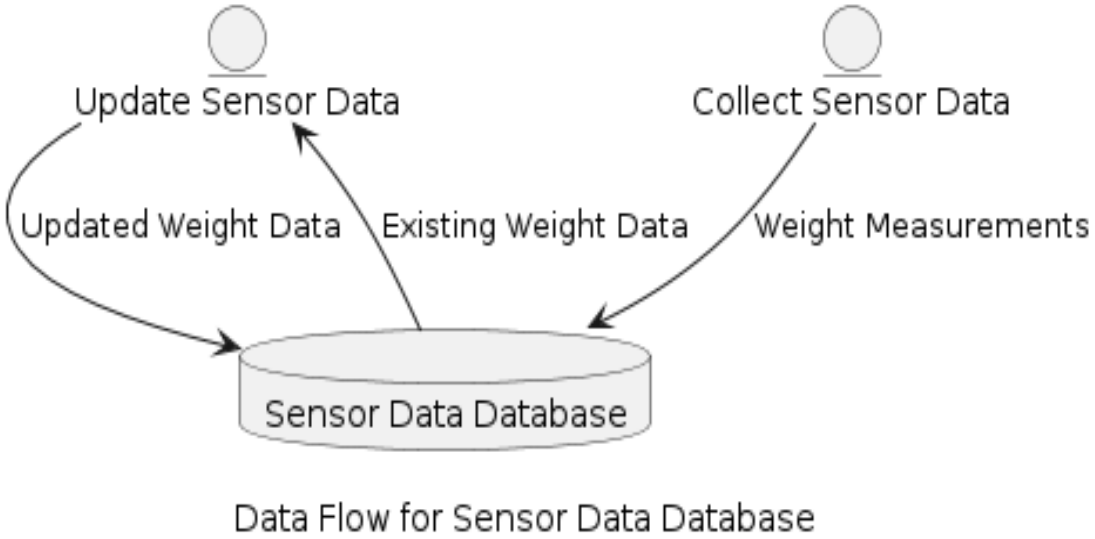


Figure 3.19: Data Flow Diagram for Weight Sensors of Smart Bin

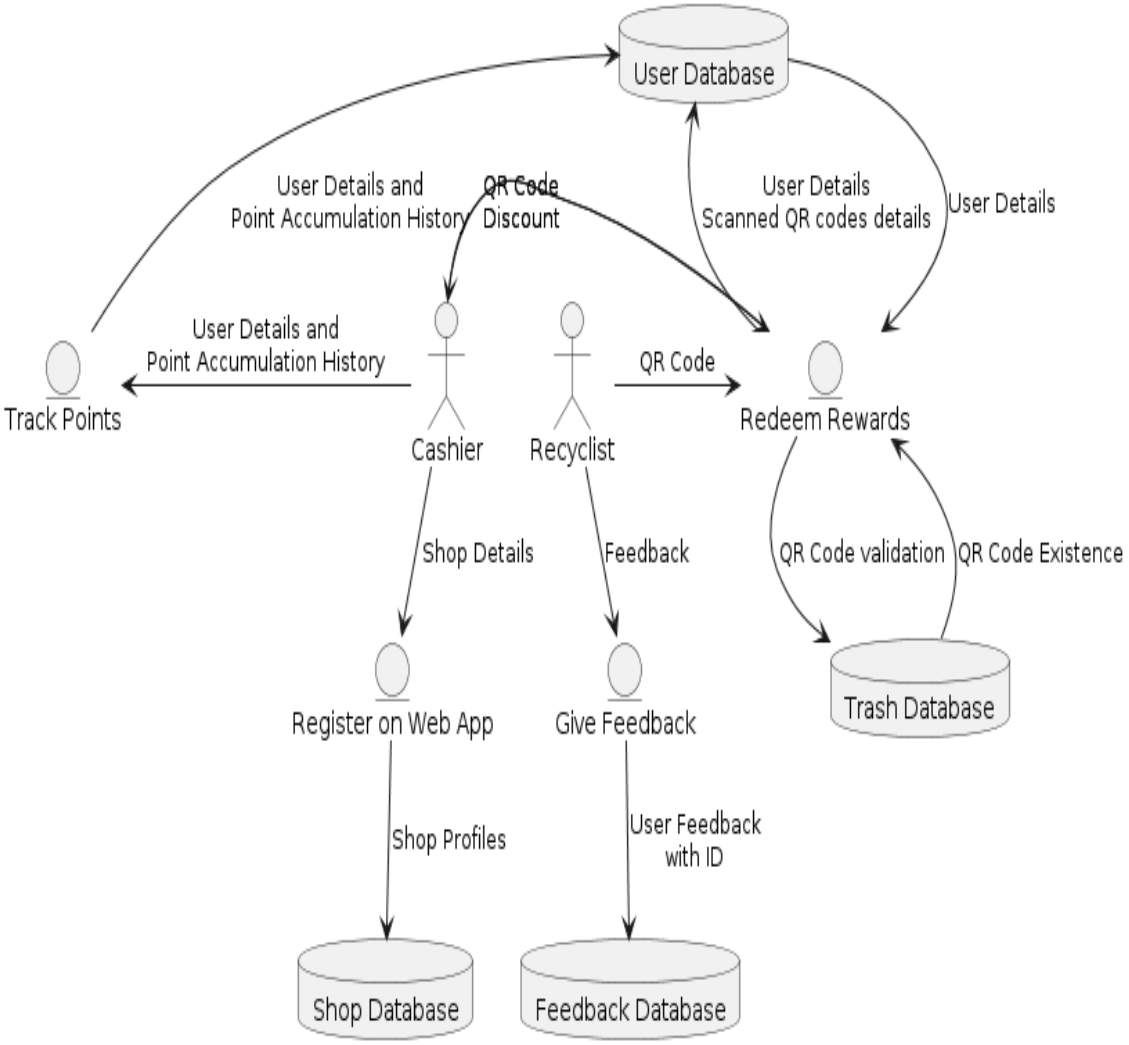


Figure 3.20: Data Flow Diagram for Feedback, Rewards, Tracking Points and Shop Registration

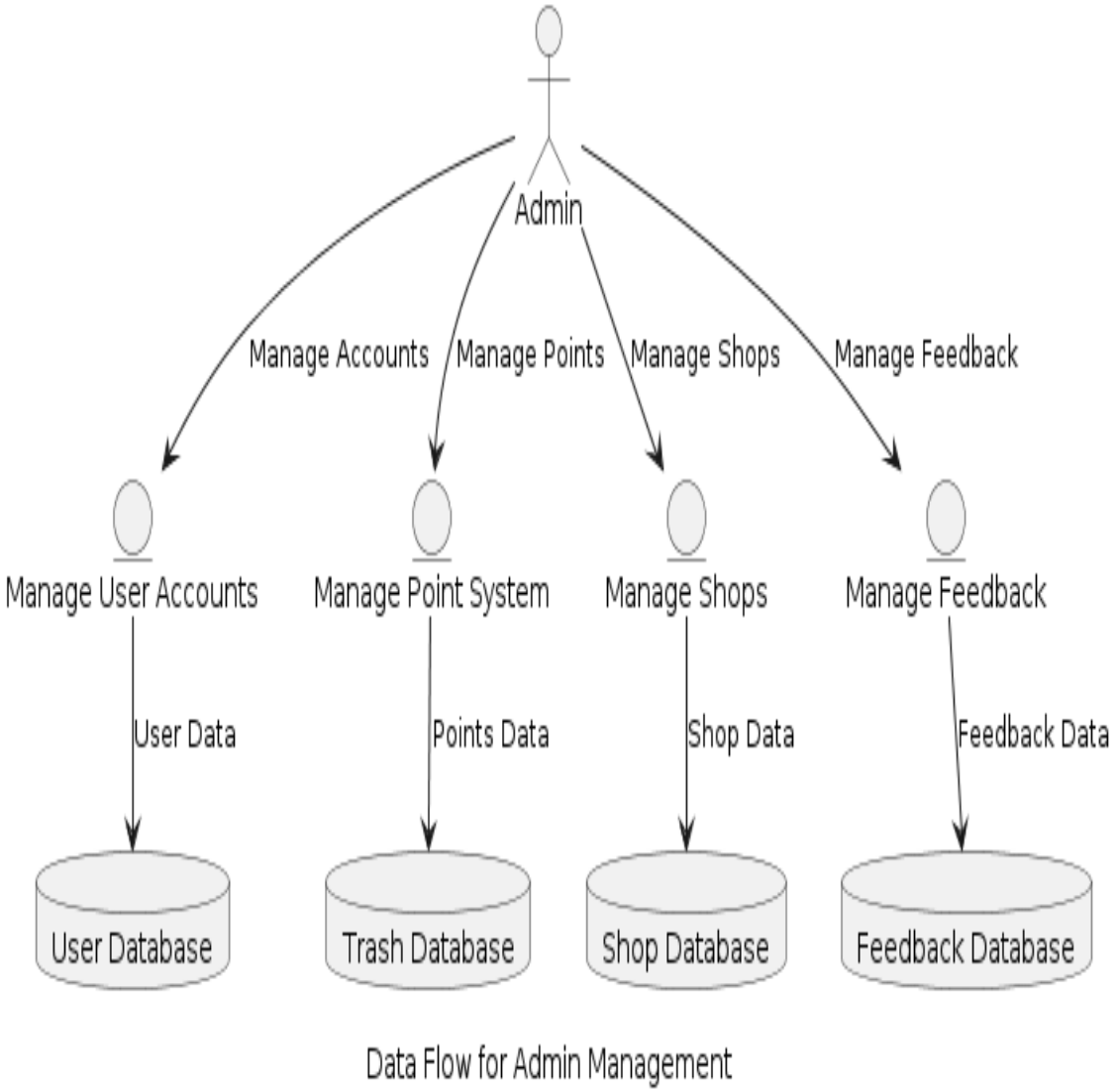


Figure 3.21: Data Flow Diagram for Admin Management

3.4 Domain Model

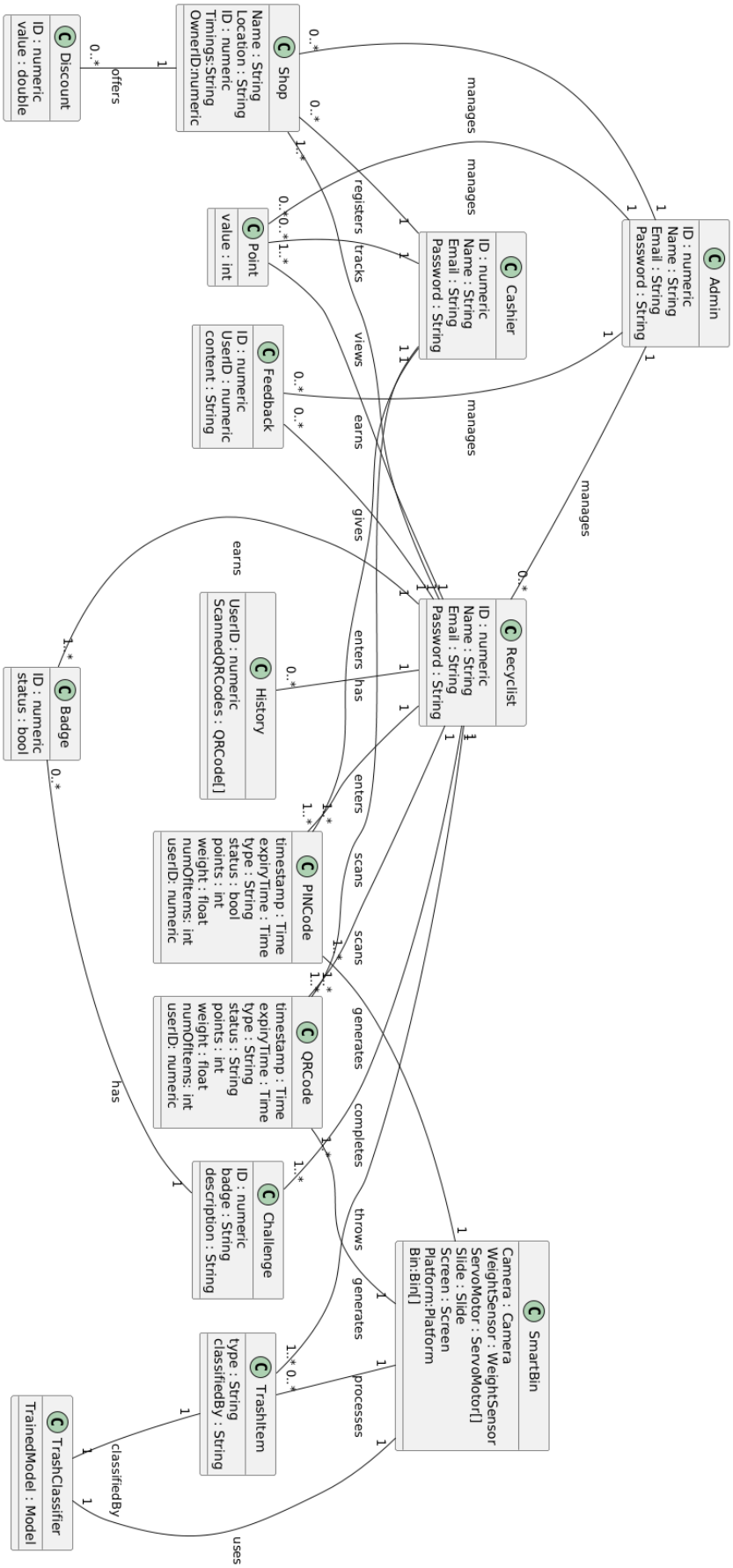


Figure 3.22: Detailed Domain Model

Chapter 4

Implementation and Testing

"Trash to Treasure" is an innovative system designed to revolutionize waste management by promoting recycling through a user-friendly, incentive-based approach. The core functionalities of the system include: smart waste classification, incentivized recycling, real-time data synchronization, user engagement, feedback mechanism.

4.1 Algorithm Design

4.1.1 AI Trash Classification Module

Listing 4.1: AI Trash Classification with Updated Outputs

```
Algorithm: AI Trash Classification

Input: image
Output: type, points, numOfItems, timestamp, expiryTime, status,
QR_Code, PIN_Code

Procedure AI_Trash_Classification(image)
    // Step 1: Load the trained YOLO model
    model <- load_YOLO_model()

    // Step 2: Detect objects in the image using YOLO
    detected_objects <- model.detect(image)

    // Step 3: Initialize variables
    type <- "Mixed Waste"
    points <- 0
    numOfItems <- 0
    timestamp <- current_time()
    expiryTime <- calculate_expiry_time(timestamp)
```

```
    status <- "active"
    QR_Code <- ""
    PIN_Code <- ""

    // Step 4: Classify detected objects
    for each object in detected_objects do
        if object.class in ["plastic", "metal", "paper"] then
            type <- object.class
            if object.class == "plastic" or object.class == "metal"
            then
                numOfItems <- number_of_items(object)
                points <- points * numOfItems
            else if object.class == "paper" then
                weight <- measure_weight(object)
                points <- points * calculate_points_from_weight(weight)
            end if
        end if
    end for

    // Step 5: Check if there are multiple objects of
    different categories
    if length(unique([obj.class for obj in detected_objects])) > 1
    then
        type <- "Mixed Waste"
        points <- 0
        numOfItems <- 0
        status <- "Invalid"
    end if

    // Step 6: Generate QR Code and PIN Code
    if type != "Mixed Waste" then
        QR_Code <- generate_QR_Code(type, points, numOfItems,
        timestamp, expiryTime, status)
        PIN_Code <- generate_PIN_Code(type, points,
        numOfItems, timestamp, expiryTime, status)
    end if

    // Step 7: Store details in database
    store_to_database(type, points, numOfItems, timestamp,
    expiryTime, status, QR_Code, PIN_Code)

    // Step 8: Return results
    return (type, points, numOfItems, timestamp,
    expiryTime, status, QR_Code, PIN_Code)
End Procedure
```

4.1.2 Smart Bin

Listing 4.2: Smart Bin Module

```
Algorithm: Smart Bin Module

Input: trash_item
Output: type, points, numOfItems, timestamp, expiryTime, status,
QR_Code, PIN_Code

Procedure Smart_Bin_Module(trash_item)
    // Step 1: Slide trash onto platform
    slide_trash_onto_platform()

    // Step 2: Capture image using camera
    image <- capture_image()

    // Step 3: Classify trash item using AI Trash Classification Algorithm
    type, points, numOfItems, timestamp,
    expiryTime, status, QR_Code, PIN_Code <- AI_Trash_Classification(image)

    // Step 4: Check if classification was successful
    if type !== "Mixed Waste" then
        rotate_slide_to_bin(type)
        open_small_door()
        drop_trash_into_bin()
        store_to_database(type, points, numOfItems,
        timestamp, expiryTime, status, QR_Code, PIN_Code)
        display_on_screen(QR_Code, PIN_Code)
        return (type, points, numOfItems,
        timestamp, expiryTime, status, QR_Code, PIN_Code)
    end if

     // Step 5: Check if classification was unsuccessful
    if type == "Mixed Waste" then
        rotate_slide_to_bin(type)
        open_small_door()
        drop_trash_into_bin()
        display_on_screen(error message)
        return ()
    end if

    // Step 5: Rotate main slide to the appropriate bin
    rotate_slide_to_bin(type)

    // Step 6: Open small door and drop trash into bin
    open_small_door()
    drop_trash_into_bin()

```

```
    // Step 7: Display QR Code and PIN Code on screen
    display_on_screen(QR_Code, PIN_Code)

    // Step 8: Store details in database
    store_to_database(type, points, numOfItems,
    timestamp, expiryTime, status, QR_Code, PIN_Code)

    // Step 9: Return results
    return (type, points, numOfItems,
    timestamp, expiryTime, status, QR_Code, PIN_Code)
End Procedure
```

4.1.3 Client Mobile App

4.1.3.1 User Registration and Profile Management

Listing 4.3: User Registration and Profile Management

```
Algorithm: User_Registration_Profile_Management

Input: user_details
Output: confirmation_message

Procedure User_Registration_Profile_Management(user_details)
    // Step 1: Capture user registration details
    user_details <- capture_user_details()

    // Step 2: Validate user input
    is_valid <- validate_user_details(user_details)

    // Step 3: Create user profile if valid
    if is_valid then
        create_user_profile(user_details)
        confirmation_message <- "Registration Successful"
    else
        confirmation_message <- "Invalid Details. Please Try Again."
    end if

    // Step 4: Return confirmation message
    return confirmation_message
End Procedure
```

Listing 4.4: Change Profile Settings

```
Algorithm: Change_Profile_Settings

Input: user_id, new_profile_data
Output: confirmation_message
```

```
5
6  Procedure Change_Profile_Settings(user_id, new_profile_data)
7      // Step 1: Fetch current user profile
8      current_profile <- fetch_user_profile(user_id)
9
10     // Step 2: Validate new profile data
11     is_valid <- validate_profile_data(new_profile_data)
12
13     // Step 3: Update profile fields individually
14     if is_valid then
15         confirmation_message <- ""
16
17         // Step 3.1: Change Password (if requested)
18         if "password" in new_profile_data then
19             if confirm_password_change(user_id,
20
21             new_profile_data.password)
22             then
23                 update_password(user_id, new_profile_data.password)
24                 confirmation_message <- confirmation_message +
25                 "Password updated successfully. "
26             else
27                 confirmation_message <- confirmation_message +
28                 "Password confirmation failed. "
29             end if
30         end if
31
32         // Step 3.2: Change First Name (if requested)
33         if "first_name" in new_profile_data then
34             update_first_name(user_id, new_profile_data.first_name)
35             confirmation_message <- confirmation_message +
36             "First name updated successfully. "
37         end if
38
39         // Step 3.3: Change Last Name (if requested)
40         if "last_name" in new_profile_data then
41             update_last_name(user_id, new_profile_data.last_name)
42             confirmation_message <- confirmation_message +
43             "Last name updated successfully. "
44         end if
45
46         // Step 3.4: Change Profile Picture (if requested)
47         if "profile_picture" in new_profile_data then
48             update_profile_picture(user_id,
49
50             new_profile_data.profile_picture)
51             confirmation_message <- confirmation_message +
52             "Profile picture updated successfully. "
```

```
        end if

        // If no changes were requested
        if confirmation_message == "" then
            confirmation_message <- "No changes were made to the profile."
        end if
    else
        confirmation_message <- "Invalid Profile Data. Please Try Again."
    end if

    // Step 4: Return confirmation message
    return confirmation_message
End Procedure
```

4.1.3.2 QR Code Scanning

Listing 4.5: QR Code Scanning

```
Algorithm: QR_Code_Scanning

Input: QR_Code
Output: points, message

Procedure QR_Code_Scanning(QR_Code)
    // Step 1: Scan QR Code
    QR_Data <- scan_QR_Code(QR_Code)

    // Step 2: Validate QR Code
    is_valid <- validate_QR_Code(QR_Data)

    // Step 3: Award points if valid
    if is_valid then
        points <- award_points(QR_Data)
        message <- "Points Awarded: " + points
    else
        points <- 0
        message <- "Invalid QR Code"
    end if

    // Step 4: Return points and message
    return (points, message)
End Procedure
```

4.1.3.3 PIN Code Entry

Listing 4.6: PIN Code Entry

```
Algorithm PIN_Code_Entry

Input: PIN_Code
Output: points, message

Procedure PIN_Code_Entry(PIN_Code)
    // Step 1: Enter PIN Code
    PIN_Data <- enter_PIN_Code(PIN_Code)

    // Step 2: Validate PIN Code
    is_valid <- validate_PIN_Code(PIN_Data)

    // Step 3: Award points if valid
    if is_valid then
        points <- award_points(PIN_Data)
        message <- "Points Awarded: " + points
    else
        points <- 0
        message <- "Invalid PIN Code"
    end if

    // Step 4: Return points and message
    return (points, message)
End Procedure
```

4.1.3.4 Real-Time Point Tracking

Listing 4.7: Real-Time Point Tracking

```
Algorithm: Real_Time_Point_Tracking

Input: user_id
Output: points

Procedure Real_Time_Point_Tracking(user_id)
    // Step 1: Fetch user points
    points <- fetch_user_points(user_id)

    // Step 2: Display real-time points
    display_points(points)

    // Step 3: Return points
    return points
End Procedure
```

4.1.3.5 Recycling Activity History

Listing 4.8: Recycling Activity History

```
Algorithm: Recycling_Activity_History

Input: user_id
Output: activity_history

Procedure Recycling_Activity_History(user_id)
    // Step 1: Fetch recycling activity history
    activity_history <- fetch_recycling_history(user_id)

    // Step 2: Display activity history
    for each record in activity_history do
        status <- record.status
        numOfItems <- record.numOfItems
        timestamp <- record.timestamp
        expiryTime <- record.expiryTime
        points <- record.points
        category <- record.type
        QR_Code <- record.QR_Code
        PIN_Code <- record.PIN_Code

        // Display details for each record
        print("Category: " + category)
        print("Points Earned: " + points)
        print("Number of Items: " + numOfItems)
        print("Timestamp: " + timestamp)
        print("Expiry Time: " + expiryTime)
        print("Status: " + status)
        print("QR Code: " + QR_Code)
        print("PIN Code: " + PIN_Code)
        print("-------------------------------")
    end for

    // Step 3: Return activity history
    return activity_history
End Procedure
```

4.1.3.6 Gamification Elements

Listing 4.9: Gamification Elements

```
Algorithm: Gamification_Elements

Input: user_id
Output: challenges, badges
```

```

Procedure Gamification_Elements(user_id)
    // Step 1: Fetch current challenges from the database
    challenges <- fetch_from_database("challenges", user_id)

    // Step 2: Fetch earned badges from the database
    badges <- fetch_from_database("badges", user_id)

    // Step 3: Display challenges
    for each challenge in challenges do
        print("Challenge ID: " + challenge.ID)
        print("Description: " + challenge.description)
        print("Status: " + challenge.status)
        print("--------------------------------")
    end for

    // Step 4: Display badges
    for each badge in badges do
        print("Badge ID: " + badge.ID)
        print("Title: " + badge.title)
        print("Description: " + challenge.description)
        print("Status: " + (if badge.status then "Earned"
        else "Not Earned"))
        print("--------------------------------")
    end for

    // Step 5: Return challenges and badges
    return (challenges, badges)
End Procedure
```

4.1.3.7 Feedback Module

Listing 4.10: Feedback Module

```
Algorithm: Feedback_Module

Input: subject, feedback, user_id
Output: confirmation_message

Procedure Feedback_Module(subject, feedback, user_id)
    // Step 1: Fetch user details from the database
    user_details <- fetch_from_database("users", user_id)

    // Step 2: Log feedback into the database
    insert_into_database("feedback", {
        "user_id": user_details.ID,
        "subject": subject,
        "feedback": feedback,
```

```
        "timestamp": current_time()
    })

    // Step 3: Return confirmation message
    confirmation_message <- "Feedback Submitted Successfully"
    return confirmation_message
End Procedure
```

4.1.3.8 View Integrated Stores

Listing 4.11: View Integrated Stores

```
Algorithm: View_Integrated_Stores

Input: user_id
Output: store_list

Procedure View_Integrated_Stores(user_id)
    // Step 1: Fetch list of integrated stores from the shop database
    store_list <- fetch_from_database("shops", {
        "filter": "integrated",
        "user_id": user_id
    })

    // Step 2: Display store list
    for each store in store_list do
        print("Store Name: " + store.Name)
        print("Location: " + store.Location)
        print("Timings: " + store.Timings)
        print("------------------------------")
    end for

    // Step 3: Return store list
    return store_list
End Procedure
```

4.1.4 Client Web App

4.1.4.1 Shop and Cafe Registration

Listing 4.12: Shop and Cafe Registration

```
Algorithm: Shop_Cafe_Registration

Input: shop_details
Output: confirmation_message
```

```

Procedure Shop_Cafe_Registration(shop_details)
    // Step 1: Capture shop registration details
    shop_details <- capture_shop_details()

    // Step 2: Validate shop details
    is_valid <- validate_shop_details(shop_details)

    // Step 3: Register shop if valid
    if is_valid then
        register_shop(shop_details)
        confirmation_message <- "Registration Successful"
    else
        confirmation_message <- "Invalid Details. Please Try Again."
    end if

    // Step 4: Return confirmation message
    return confirmation_message
End Procedure
```

4.1.4.2 QR Code Scanning

Listing 4.13: QR Code Scanning with Database Validation

```
Algorithm: QR_Code_Scanning_Web

Input: QR_Code
Output: points, message

Procedure QR_Code_Scanning_Web(QR_Code)
    // Step 1: Fetch QR Code details from database
    QR_Data <- fetch_from_database("qrcodes", {
        "code": QR_Code
    })

    // Step 2: Validate QR Code
    if QR_Data is NULL then
        points <- 0
        message <- "Invalid QR Code"
        return (points, message)
    end if

    if QR_Data.status != "active" then
        points <- 0
        message <- "QR Code status is " + QR_Data.status
        return (points, message)
    end if

```

```
    // Step 3: Award points and update status
    points <- QR_Data.points
    update_database("qrcodes", {
        "code": QR_Code,
        "status": "redeemed"
    })
    update_user_points(QR_Data.userID, points)
    message <- "Points Awarded: " + points

    // Step 4: Return points and message
    return (points, message)
End Procedure
```

4.1.4.3 PIN Code Entry and Authentication

Listing 4.14: PIN Code Entry and Authentication with Database Validation

```
Algorithm: PIN_Code_Entry_Auth

Input: PIN_Code
Output: points, message

Procedure PIN_Code_Entry_Auth(PIN_Code)
    // Step 1: Fetch PIN Code details from database
    PIN_Data <- fetch_from_database("pincodes", {
        "code": PIN_Code
    })

    // Step 2: Validate PIN Code
    if PIN_Data is NULL then
        points <- 0
        message <- "Invalid PIN Code"
        return (points, message)
    end if

    if PIN_Data.status != "active" then
        points <- 0
        message <- "PIN Code status is " + PIN_Data.status
        return (points, message)
    end if

    // Step 3: Award points and update status
    points <- PIN_Data.points
    update_database("pincodes", {
        "code": PIN_Code,
        "status": "redeemed"
    })
    update_user_points(PIN_Data.userID, points)
```

```
    message <- "Points Awarded: " + points

    // Step 4: Return points and message
    return (points, message)
End Procedure
```

4.1.4.4 Update Cashier Settings

Listing 4.15: Update Cashier Settings via Web App

```
Algorithm: Update_Cashier_Settings

Input: cashier_id, new_settings
Output: confirmation_message

Procedure Update_Cashier_Settings(cashier_id, new_settings)
    // Step 1: Fetch current cashier profile
    current_profile <- fetch_from_database("cashiers", {
        "id": cashier_id
    })

    // Step 2: Validate new settings
    is_valid <- validate_settings(new_settings)
    if not is_valid then
        return "Invalid settings. Please check your input."
    end if

    // Step 3: Update settings individually
    confirmation_message <- ""

    // Step 3.1: Change Name
    if "name" in new_settings then
        update_database("cashiers", {
            "id": cashier_id,
            "name": new_settings.name
        })
        confirmation_message <- confirmation_message +
        "Name updated successfully. "
    end if

    // Step 3.2: Change Password
    if "password" in new_settings then
        if confirm_password_change(cashier_id, new_settings.password) then
            update_database("cashiers", {
                "id": cashier_id,
                "password": hash_password(new_settings.password)
            })
            confirmation_message <- confirmation_message +
```

```
            "Password updated successfully. "
        else
            confirmation_message <- confirmation_message +
            "Password confirmation failed. "
        end if
    end if

    // Step 3.3: Change Profile Picture
    if "profile_picture" in new_settings then
        update_database("cashiers", {
            "id": cashier_id,
            "profile_picture": new_settings.profile_picture
        })
        confirmation_message <- confirmation_message +
        "Profile picture updated successfully. "
    end if

    // Step 4: Final confirmation message
    if confirmation_message == "" then
        confirmation_message <- "No changes were made."
    end if

    // Step 5: Return confirmation message
    return confirmation_message
End Procedure
```

4.1.5 Admin Web App

4.1.5.1 User Account Management

Listing 4.16: User Account Management

```
Algorithm: User_Account_Management

Input: admin_credentials, action, user_details
Output: confirmation_message

Procedure User_Account_Management(admin_credentials, action, user_details)
    // Step 1: Authenticate admin
    is_authenticated <- authenticate_admin(admin_credentials)

    // Step 2: Perform action if authenticated
    if is_authenticated then
        switch (action)
            case "create":
                create_user(user_details)
                confirmation_message <- "User Created Successfully"
```

```
            case "update":
                update_user(user_details)
                confirmation_message <- "User Updated Successfully"
            case "delete":
                delete_user(user_details.user_id)
                confirmation_message <- "User Deleted Successfully"
            case "view":
                user_info <- view_user(user_details.user_id)
                confirmation_message <- user_info
            otherwise:
                confirmation_message <- "Invalid Action"
        end switch
    else
        confirmation_message <- "Authentication Failed"
    end if

    // Step 3: Return confirmation message
    return confirmation_message
End Procedure
```

4.1.5.2 Point System Management

Listing 4.17: Point System Management

```
Algorithm: Point_System_Management

Input: admin_credentials, action, point_data
Output: confirmation_message

Procedure Point_System_Management(admin_credentials, action, point_data)
    // Step 1: Authenticate admin
    is_authenticated <- authenticate_admin(admin_credentials)

    // Step 2: Perform action if authenticated
    if is_authenticated then
        switch (action)
            case "adjust":
                adjust_point_values(point_data)
                confirmation_message <- "Point Values
                Adjusted Successfully"
            case "set_rewards":
                set_rewards(point_data)
                confirmation_message <- "Rewards Set Successfully"
            case "monitor":
                point_distribution <- monitor_point_distribution()
                confirmation_message <- point_distribution
            otherwise:
                confirmation_message <- "Invalid Action"
```

```
        end switch
    else
        confirmation_message <- "Authentication Failed"
    end if

    // Step 3: Return confirmation message
    return confirmation_message
End Procedure
```

4.1.5.3 Shop and Cafe Management

Listing 4.18: Shop and Cafe Management

```
Algorithm: Shop_Cafe_Management

Input: admin_credentials, action, shop_data
Output: confirmation_message

Procedure Shop_Cafe_Management(admin_credentials, action, shop_data)
    // Step 1: Authenticate admin
    is_authenticated <- authenticate_admin(admin_credentials)

    // Step 2: Perform action if authenticated
    if is_authenticated then
        switch (action)
            case "update":
                update_shop_info(shop_data)
                confirmation_message <- "Shop Information Updated"
            case "delete":
                delete_shop(shop_data.shop_id)
                confirmation_message <- "Shop Deleted"
            case "update":
                update_shop(shop_data.shop_id)
                confirmation_message <- "Shop Updated"
            case "view":
                view_shop(shop_data.shop_id)
                confirmation_message <- view_shop
            otherwise:
                confirmation_message <- "Invalid Action"
        end switch
    else
        confirmation_message <- "Authentication Failed"
    end if

    // Step 3: Return confirmation message
    return confirmation_message
End Procedure
```

4.1.5.4 Feedback Management

Listing 4.19: Feedback Management

```
Algorithm: Feedback_Management

Input: admin_credentials, action, feedback_data
Output: confirmation_message

Procedure Feedback_Management(admin_credentials, action, feedback_data)
    // Step 1: Authenticate admin
    is_authenticated <- authenticate_admin(admin_credentials)

    // Step 2: Perform action if authenticated
    if is_authenticated then
        switch (action)
            case "view":
                feedback_list <- view_feedback()
                confirmation_message <- feedback_list
            otherwise:
                confirmation_message <- "Invalid Action"
        end switch
    else
        confirmation_message <- "Authentication Failed"
    end if

    // Step 3: Return confirmation message
    return confirmation_message
End Procedure
```

4.2 External APIs/SDKs

API and Version	Description	Purpose of Usage	API Endpoint/Function/Class Used
Ultralytics YOLO	A Python library for YOLO-based object detection and classification. YOLO (You Only Look Once) is a state-of-the-art, real-time object detection system. YOLOv8 is the latest version, providing improved accuracy and speed for object detection tasks.	Used for detecting and classifying types of trash in the AI-based Trash Classification Module.	`model <- load_YOLO_model()` `detected_objects <- model.detect(image)`

Firebase SDK	Firebase SDK is widely used for integrating various backend functionalities without requiring custom server-side code. Firebase Storage is used to upload and manage files (e.g., profile pictures, receipts). Firestore is used to store structured data like trash classification history, user points, and QR/PIN codes. Firebase Authentication is used to manage user authentication seamlessly.	Backend-as-a-Service (BaaS) for various functionalities like storage, authentication, and database.	`store_to_database()`
			`create_user_profile()`
			`update_profile_picture()`
			`update_last_name()`
			`update_first_name()`
			`update_password()`
			`validate_QR_Code()`
			`validate_PIN_Code()`
			`fetch_user_points()`
			`fetch_recycling_history()`
			`fetch_from_database()`
			`insert_into_database()`
			`register_shop()`
			`create_user()`
			`update_user()`
			`adjust_point_values()`
			`update_shop_info()`
			`view_feedback()`

Table 4.1: External SDKs and APIs

4.3 Testing Details

4.3.1 Unit Testing

4.3.1.1 Test Cases

1. User Registration

Test Case ID	TC-1.1
Written By	Manahil Shakeel
Test Suite	User Registration
Test Type	Manual
Objective	Verify that users can register and set up their profiles correctly.
Execution Steps	1. Open the app. 2. Navigate to the registration page. 3. Enter full name. 4. Enter correct email address. 5. Enter an 8 digit password 6. Click next. 7. Enter correct 4 digit OTP sent by the system on mobile number. 8. Submit Form
Pre-Conditions	None (user is not logged in).
Post Conditions	User account is created, and user is logged in
Input data/Events	User's full name, email address, mobile number and a new password
Expected Output data/Events	Account is created, and user receives a welcome message.
Actual Results	As Expected.
Pass/Fail	Pass

2. Scan QR Code

Test Case ID	TC-1.2
Written By	Manahil Shakeel
Test Suite	Scan QR Code
Test Type	Manual
Objective	Verify that users can scan QR codes correctly and receive the appropriate response.
Execution Steps	1. Open the app. 2. Navigate to the QR code scanner. 3. Scan the displayed QR code
Pre-Conditions	1. User is logged in to the system. 2. QR code is displayed and accessible.
Post Conditions	User receives reward based on the information stored in the QR code scanned.
Input data/Events	Mobile phone camera permissions
Expected Output data/Events	QR code is scanned successfully and is stored in database.
Actual Results	As Expected.
Pass/Fail	Pass

3. Throw Trash in Bin

Test Case ID	TC-1.2
Written By	Manahil Shakeel
Test Suite	Throw Trash in Bin
Test Type	Manual
Objective	Verify that the system categorizes the trash correctly when thrown into the bin.
Execution Steps	1. Throw trash into opening of the bin.
Pre-Conditions	1. User is logged in ti the system. 2. Bin is operational and connected to the system..
Post Conditions	Trash is categorized and recorded.
Input data/Events	Trash being thrown into the bin.
Expected Output data/Events	System identifies the type of trash from the 4 categories (plastic, paper, metal, other wastes) and and disposed off in designated bin.
Actual Results	As Expected.
Pass/Fail	Pass

4. View History

Test Case ID	TC-1.2
Written By	Manahil Shakeel
Test Suite	View History
Test Type	Manual
Objective	Verify that users can view their redemption and recycling history correctly.
Execution Steps	1. Open the app. 2. Navigate to the history section.
Pre-Conditions	User is logged in to the system.
Post Conditions	User views their history.
Input data/Events	Navigation to the history section.
Expected Output data/Events	Display of user's redemption and recycling history.
Actual Results	As Expected.
Pass/Fail	Pass

5. Complete Challenges

Test Case ID	TC-1.2
Written By	Manahil Shakeel
Test Suite	Complete Challenges
Test Type	Manual
Objective	Verify that users can complete challenges and receive rewards.
Execution Steps	1. Open the app. 2. Navigate to the challenges section. 3. Complete a listed challenge. 4. Submit the challenge.
Pre-Conditions	User is logged in and challenges are available.
Post Conditions	1. Challenge is marked as completed and the badges are rewarded to the user. 2. User's level is increased
Input data/Events	Challenge completion data.
Expected Output data/Events	Badges are added to user's account.
Actual Results	As Expected.
Pass/Fail	Pass

6. Redeem Rewards

Test Case ID	TC-1.2
Written By	Manahil Shakeel
Test Suite	Redeem Rewards
Test Type	Manual
Objective	Verify that users can redeem rewards correctly based on their points.
Execution Steps	1. Purchase something from the partnered shop. 2. According to the information in the user QR code or the pin code, the user's points are deducted accordingly.
Pre-Conditions	User is logged in and has sufficient points.
Post Conditions	Points are deducted, and reward is redeemed from the partnered shop.
Input data/Events	Providing QR code or pin code to the cashier.
Expected Output data/Events	Points are deducted from user's account, the database is updated and the purchase is made successfully.
Actual Results	As Expected.
Pass/Fail	Pass

7. View Integrated Shops

Test Case ID	TC-1.2
Written By	Manahil Shakeel
Test Suite	View Integrated Shops
Test Type	Manual
Objective	Verify that users can view the list of integrated shops.
Execution Steps	1. Open the app. 2. Navigate to the integrated shops section.
Pre-Conditions	User is logged in and integrated shops are configured.
Post Conditions	List of integrated shops is displayed.
Input data/Events	Navigation to integrated shops section.
Expected Output data/Events	Display of the list of integrated shops.
Actual Results	As Expected.
Pass/Fail	Pass

8. Give User Feedback

Test Case ID	TC-1.2
Written By	Manahil Shakeel
Test Suite	Give User Feedback
Test Type	Manual
Objective	Verify that users can submit feedback and comments.
Execution Steps	1. Open the app. 2. Navigate to the feedback section. 3. Enter feedback and comments. 4. Submit feedback.
Pre-Conditions	User is logged in.
Post Conditions	Feedback is recorded.
Input data/Events	Feedback and comments.
Expected Output data/Events	Feedback is successfully submitted and a confirmation is displayed.
Actual Results	As Expected.
Pass/Fail	Pass

9. Web Application Registration

Test Case ID	TC-1.1
Written By	Manahil Shakeel
Test Suite	Web Application Registration
Test Type	Manual
Objective	Verify that shop cashiers can register and set up their profiles correctly.
Execution Steps	1. Open the web app. 2. Navigate to the registration page. 3. Enter full name. 4. Enter correct email address. 5. Enter an 8 digit password 6. Click next. 7. Enter correct 4 digit OTP sent by the system on mobile number. 8. Submit Form
Pre-Conditions	None (cashier is not logged in).
Post Conditions	1. Shop is integrated with the system and the shop account is created. 2. The cashier is logged in
Input data/Events	Shop's full name, valid email address, mobile number and a new password
Expected Output data/Events	Account is created, and cashier receives a welcome message.
Actual Results	As Expected.
Pass/Fail	Pass

10. Scan QR Code (for web app)

Test Case ID	TC-1.2
Written By	Manahil Shakeel
Test Suite	Scan QR Code (for web app)
Test Type	Manual
Objective	Verify that users can scan QR codes correctly and receive the appropriate response.
Execution Steps	1. Open the app. 2. Navigate to the QR code scanner. 3. Scan the displayed QR code by user.
Pre-Conditions	1. Cashier is logged in to the system. 2. QR code is displayed and accessible.
Post Conditions	User's information is displayed on cashier's screen.
Input data/Events	Device's camera permissions
Expected Output data/Events	1. The QR code is scanned successfully and user's points information is displayed to the cashier. 2. The QR code's validity is expired and it cannot be used later on.
Actual Results	As Expected.
Pass/Fail	Pass

11. Track User Points

Test Case ID	TC-1.2
Written By	Manahil Shakeel
Test Suite	Track User Points
Test Type	Manual
Objective	Verify that shop keepers can track points and provide discounts.
Execution Steps	1. Open web app. 2.The cashier scans user's QR code or enters the pin code provided by user into his/her system. 3. Cashier can view the existing user points in their wallet after scanning the QR code/pin code provided. 4. After providing the discounts, the cashier views user's remaining points on their system and verbally confirms with the buyer.
Pre-Conditions	1. Cashier is logged in to the system. 2. The QR code scanned or the pin code entered is valid with correct information displayed.
Post Conditions	Points and corresponding discounts are displayed.
Input data/Events	QR code or pin code
Expected Output data/Events	Display of points and calculated discounts.
Actual Results	As Expected.
Pass/Fail	Pass

12. Manage User Accounts

Test Case ID	TC-1.2
Written By	Manahil Shakeel
Test Suite	Manage User Accounts
Test Type	Manual
Objective	Verify that admins can manage user accounts effectively.
Execution Steps	1. Open the admin panel. 2. Navigate to the user management section. 3. Create a new user account. 4. Approve or delete existing accounts.
Pre-Conditions	Admin is logged in.
Post Conditions	User accounts are created, approved, or deleted.
Input data/Events	User's name, User's account ID allocated
Expected Output data/Events	Admin is able to manage user accounts efficiently as specified.
Actual Results	As Expected.
Pass/Fail	Pass

13. Manage Integrated Shops

Test Case ID	TC-1.2
Written By	Manahil Shakeel
Test Suite	Manage Integrated Shops
Test Type	Manual
Objective	Verify that admins can manage shop accounts effectively.
Execution Steps	1. Open the admin panel. 2. Navigate to the shop management section. 3. Create a new shop account. 4. Delete existing shop accounts if needed.
Pre-Conditions	Admin is logged in.
Post Conditions	Shop accounts are created or deleted as specified.
Input data/Events	Shop's registered names, Shop ID allocated.
Expected Output data/Events	Admin is able to manage shop accounts efficiently as specified.
Actual Results	As Expected.
Pass/Fail	Pass

14. Manage Points System

Test Case ID	TC-1.2
Written By	Manahil Shakeel
Test Suite	Manage Points System
Test Type	Manual
Objective	Verify that admins can configure and manage the points system correctly.
Execution Steps	1. Open the admin panel. 2. Navigate to the points management section. 3. Set thresholds and values for trash categories.
Pre-Conditions	Admin is logged in.
Post Conditions	Points system is updated as configured.
Input data/Events	Threshold and category values.
Expected Output data/Events	Points system is configured and applied correctly.
Actual Results	As Expected.
Pass/Fail	Pass

15. Manage User Feedback

Following is the example of Unit testing:

Test case ID	Test Objective	Precondition	Steps	Test data	Expected result	Post-condition	Actual Result	Pass/fail
TC001	Verify admin login with username and password	Admin should be registered with valid email and password before login.	Click on Login button Enter valid username and password	Email-id: abc@xyz.com Password: Xyz123	System displays Admin homepage	Admin should be kept logged in until logout.	As Expected,	Pass

Figure 4.1: Example for Unit Testing

Test Case ID	TC-1.2
Written By	Manahil Shakeel
Test Suite	Manage User Feedback
Test Type	Manual
Objective	Verify that admins can manage user feedback effectively.
Execution Steps	1. Open the admin panel. 2. Navigate to the feedback management section. 3. View and delete feedback as needed. 4. Admin can reply to any user comments if required.
Pre-Conditions	Admin is logged in.
Post Conditions	Feedback is managed and deleted if necessary.
Input data/Events	Feedback details, Replies to user comments.
Expected Output data/Events	Feedback is viewed, replied to and managed correctly.
Actual Results	As Expected.
Pass/Fail	Pass

Chapter 5

Conclusions and Future Work

5.1 Conclusion

The "Trash to Treasure" project represents a comprehensive and innovative solution to the critical issue of waste management and recycling. Over the course of a year, we successfully designed and implemented an AI-powered trash classification and reward system that integrates cutting-edge technologies, including YOLO-based object detection, a smart bin equipped with IoT components, and user-centric mobile and web applications. The project effectively incentivizes recycling through a points-based rewards system, enabling users to earn and redeem points for discounts while fostering sustainable waste disposal behaviors. Key features such as real-time trash classification, QR and PIN code generation, recycling activity tracking, and gamification elements like challenges and badges, create a seamless and engaging user experience. The smart bin's ability to classify trash into plastic, metal, and paper categories with automated mechanisms further highlights the system's practicality and innovation. By integrating Firebase for data storage and authentication, and leveraging third-party APIs for efficient system functionality, "Trash to Treasure" sets a strong foundation for scalable and impactful waste management solutions. This project not only demonstrates technical excellence but also underscores the importance of promoting environmental sustainability through technology-driven initiatives.

5.2 Future Work

5.2.1 Advanced AI Techniques for Broader Waste Classification

- **Current Limitations:** Our system is limited to paper, plastic, and metal.

- **Future Work:** Research into deep learning models such as convolutional neural networks (CNNs) with transfer learning can improve classification accuracy and enable the system to identify additional waste types like glass, organic waste, and hazardous materials.

- **Methodology:** Our model, YOLOv8 could be trained on larger datasets to recognize diverse waste categories, even in poor lighting or mixed conditions. Incorporating multi-spectral imaging could enhance the recognition of non-visible features (e.g., moisture content, chemical composition).

- **Research Value:** Expanding capabilities will increase the applicability of our system in broader contexts, such as municipal or industrial waste management.

5.2.1.1 Integration with IoT for Smart Cities

- **Current Limitation:** The project does not address full-scale integration into urban systems.

- **Future Work**: Embedding IoT-enabled sensors for monitoring bin capacity, air quality, and location tracking can provide real-time data. Integrating this data with municipal waste management systems can be done for automated collection scheduling and route optimization.

- **Tools**: IoT platforms like AWS IoT Core, Azure IoT Hub, or open-source solutions like Node-RED can be explored.

- **Research Value**: This would position our system as part of smart city infrastructure, promoting sustainable urban living and reducing operational inefficiencies.

5.2.2 Integrating third-party payment APIs

- **Current Limitation:** There is no payment API integrated which does not ensure safe and secure transactions.

- **Future Work:** Integrating third-party payment APIs into the smart bin system can provide a secure and seamless method for users to redeem points for discounts.

- **Tools:** Payment APIs like PayPal, Stripe, and local gateways such as JazzCash or Easypaisa provide secure, encrypted transactions, ensuring user payment data is protected (Stripe, PayPal, JazzCash)

5.2.3 Integration with IoT for Smart Cities

- **Current Limitation**: The project does not address full-scale integration into urban systems.

- **Future Work**: Embedding IoT-enabled sensors for monitoring bin capacity, air quality, and location tracking can provide real-time data. Integrate this data with municipal waste management systems for automated collection scheduling and route optimization.

- **Tools**: IoT platforms like AWS IoT Core, Azure IoT Hub, or open-source solutions like Node-RED can be explored.

- **Research Value**: This would position our system as part of smart city infrastructure, promoting sustainable urban living and reducing operational inefficiencies.

5.2.4 Hardware Enhancements

- **Current Limitaion:** The existing hardware implementation of your project involves an open structure with a basket for trash disposal and a loosely mounted camera above it.

5.2.4.1 Closed Bin Structure

- **Current Limitation**: The open structure lacks the appearance and functionality of a traditional trash bin. It exposes the trash and internal components, which may not appeal to users or align with environmental standards.

- **Future Work**: Develop a fully enclosed bin with a proper lid to resemble a traditional waste bin. This closed design will help:

    1. Prevent external interference, such as tampering with the camera or disposed items.
    2. Contain odors and improve hygiene by isolating the waste inside.
    3. Create a cleaner and more professional aesthetic, making the bin suitable for public and commercial spaces.

- **Material Choice**: Use durable, lightweight materials such as aluminum or high-density plastic to ensure sturdiness without compromising portability.

5.2.4.2 Camera Fixation and Positioning

- **Current Limitation**: The camera above the basket is not properly fixed, potentially affecting detection accuracy due to misalignment or instability.
- **Future Work**:
  - Use an adjustable mount for the camera to fix its position securely and allow fine-tuning during calibration.
  - Consider an enclosed camera compartment within the bin structure to protect it from dust, moisture, or accidental damage.
  - Position the camera at an optimal angle and height to ensure clear visibility of trash items regardless of their size or placement.

5.2.4.3 Lighting for Night-Time Detection

- **Current Limitation**: The system does not function effectively in low-light conditions, as it lacks a dedicated lighting mechanism.
- **Future Work**:
  - Integrate LED lighting within the bin to illuminate the area under the camera.
  - The lighting system could be activated only when the bin is in use to conserve energy. This can be achieved using motion or proximity sensors that detect when an item is placed in the basket.
  - Use a diffuser or soft light mechanism to avoid harsh reflections that could interfere with the camera's image processing.
  - Ensure the light source is synchronized with the camera for consistent image capture.

5.2.4.4 Enhanced Usability Features

- **Interactive Lid Mechanism**: Add an automated lid controlled by a motor or servo. The lid could open only when the bin is ready to accept waste, further enforcing cleanliness and order.
- **Internal Bin Design**: Use color-coded smaller bins inside to align with the waste categories (plastic, paper, metal, mixed). This visual cue can enhance understanding and ensure better segregation.
- **Display Integration**: Replace the laptop with a **dedicated LCD or OLED screen** for displaying QR codes, classification results, and user instructions.

5.2.4.5 Environmental and Cost Considerations

- **Sustainability**: Use recycled or eco-friendly materials for the bin's outer structure to align with the project's recycling goals.
- **Cost-Effective Scaling**: Ensure that the enhancements remain within budget constraints and can be scaled for deployment in multiple locations.

Bibliography

[1] Mostafa Abla. Garbage classification dataset. https://www.kaggle.com/datasets/mostafaabla/garbage-classification, 2024. Accessed: 2023-02-13.

[2] Bin-e. Bin-e: Smart waste bin. https://bine.world/solutions-bine. Accessed: 2023-05-13.

[3] CleanRobotics. 4 surprisingly awesome smart bin alternatives to recycling bins. https://cleanrobotics.com/4-surprisingly-awesome-smart-bin-alternatives-to-rec Accessed: 2023-05-13.

[4] CleanRobotics. Trashbot. https://cleanrobotics.com/trashbot/. Accessed: 2023-06-03.

[5] MyMatR Corp. Mymatr: Smart waste bin for efficient waste management, 2023. Accessed: 2024-12-01.

[6] Waste Dive. Oscar pocket: Ai-powered sorting solution, 2023. Accessed: 2024-12-01.

[7] EvoEco. Evoeco. https://evoeco.com/index.html. Accessed: 2023-05-13.

[8] Institute of Policy Studies, Pakistan. How can pakistan reduce plastic pollution? https://iips.com.pk/how-can-pakistan-reduce-plastic-pollution/, 2023. Accessed: 2023-02-13.

[9] Intuitive AI. Oscar sort. https://intuitiveai.ca/oscar-sort. Accessed: 2023-05-13.

[10] WholeSum KY. Oscar sort at the university of kentucky: Promoting recycling through rewards, 2023. Accessed: 2024-12-01.

[11] Shiza Malik. Plastic crisis. https://www.dawn.com/news/1505436, 2019. Accessed: 2023-02-13.

[12] Shahid Sattar and Noreen Akhtar. Our planet is choking on plastic. https://aptma.org.pk/our-planet-is-choking-on-plastic/, 2023. Accessed: 2023-03-13.

[13] Jamal Shahid. Only 3pc of plastic is recycled in pakistan: Unep head. https://www.dawn.com/news/1734827, 2023. Accessed: 2023-02-13.

[14] U.S. Department of Commerce, International Trade Administration. Pakistan waste management. https://www.trade.gov/country-commercial-guides/pakistan-waste-management, 2024. Accessed: 2023-03-13.