6-14-2024

MonteRecycle: An Application for Streamlining Tracking of Recyclables in Uruguay

Aayooshi Dharmadhakari

Danny Walsh

Eason Ke

Kamya Krishnan

SANTA CLARA UNIVERSITY

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Date: June 14, 2024

I HEREBY RECOMMEND THAT THE THESIS PREPARED UNDER MY SUPERVISION BY

Aayooshi Dharmadhakari
Danny Walsh
Eason Ke
Kamya Krishnan

ENTITLED

MonteRecycle: An Application for Streamlining Tracking of Recyclables in Uruguay

BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

BACHELOR OF SCIENCE IN WEB DESIGN AND ENGINEERING

Shiva Jahangiri

Thesis Advisor

[illegible]

Department Chair

MonteRecycle: An Application for Streamlining Tracking of Recyclables in Uruguay

by

Aayooshi Dharmadhakari
Danny Walsh
Eason Ke
Kamya Krishnan

Submitted in partial fulfillment of the requirements
for the degree of
Bachelor of Science in Web Design and Engineering
School of Engineering
Santa Clara University

Santa Clara, California
June 14, 2024

MonteRecycle: An Application for Streamlining Tracking of Recyclables in Uruguay

Aayooshi Dharmadhakari
Danny Walsh
Eason Ke
Kamya Krishnan

Department of Computer Science and Engineering
Santa Clara University
June 14, 2024

ABSTRACT

Recycling is a crucial process that has a positive impact on nature, cleans the environment, and saves time and money. Classifying recyclables by material and recycling these items accordingly helps make our world more environmentally-friendly and a better place to live.

In the recycling system in Montevideo, Uruguay, third-party contractors hired by San Vicente transport clients' recyclables to a central recycling center or to a person that does recycling. In the past, San Vicente has transported recyclables though horse and buggy transportation. However, there has been a push from the city to replace the horses with motorcycles to reduce animal waste. As the first approach, San Vicente used pen and paper for tracking the type, weight, and pickup location of the received materials. Losing the paper on the field due to the nature of the job and rain, lack of a standard template for collecting the data, as well as its lack of error-handling features are some of the downsides of the pen and paper approach. Simplicity of using pen and paper, however, is considered as the pros of this approach.

To improve this process, San Vicente tried using Google Forms for collecting information and storing them into spreadsheets which could then later be used by the admins to verify the data and generate monthly reports for their clients. While this approach alleviates the challenges of losing papers, its usability is more complex for the field workers with limited literature, math, and technological skills.

To solve this problem, we made a full-stack application that tracks recycling data and reports accurate pickup totals and classification details to clients. We designed and developed MonteRecycle using React Native (for the UI), Postgres (for strong data), React JS (for the admin web application), and Google Cloud (for hosting the Postgres). Additionally, we developed the admin website to display all of the data gathered in the mobile application to the administrator of this recycling process. Our project is successfully implemented, tested, and provided as a downloadable Android APK to San Vicente to be used on the field as their main framework for entering and organizing data as well as generating monthly reports for San Vicente's clients. Through this project, we learned that change is inevitable in the software development process and that user feedback is the most important feedback we can get to improve our application.

Table of Contents

List of Figures

Chapter 1

Introduction

1.1 Problem Statement

Uruguay is currently developing recycling systems to help reuse and reduce waste. A company in Montevideo, Uruguay, called San Vicente, has asked the Frugal Innovation Hub to design a mobile application and website to help track recycling data and generate reports for the organization administrator and their clients. San Vicente's current recycling process consists of three main roles: the receptor, the classifier, and the administrator. The receptor role is in charge of collecting and counting bags at the pickup site. They are responsible for reporting the number of bags and what color they are which indicates what material may be in there. Optionally, the receptor can give a bag a certain weight. The classifier is responsible for classification of all materials received. They split them into categories such as paper, plastic, cardboard, metal, and other materials. They separate recyclables from non-recyclables and record the weights in order to accurately compensate the clients. The admin role controls the whole system. They have the ability to get all clients, get client details, add clients, edit clients, and delete clients. The admin also has the ability to get all users, get user details, add users, edit users, and delete users. The admin will have access to the client list page, the client detail page, the data review page, the user list page, the user detail page, the data review page, and the monthly report feature. Therefore, San Vicente will use this app to classify the recyclables collected at the pickup site by their type, weight, and pickup location, and report the number of kilograms of each material that the client recycled in a certain month.

Uruguay needs a more effective, accurate, organized, and secure data collection and analysis system for their recycling process rather than their current paper-based approach. To improve on their paper-based approach, the San Vicente administrators switched to using Google Forms for data collection. However, Google Forms was not capable of providing notifications in case of an error in the data, a feature that MonteRecycle addresses in multiple places in the app. Moreover, San Vicente needs a robust tool that will better help their clients understand where clients' recyclables are coming from. Upgrading San Vicente's tracking system for recyclables will greatly help San Vicente's clients and Uruguay's environment. MonteRecycle has a two-fold process: first, track the data that the receptor, classifier, and administrator record, and second, analyze that data. Through this streamlined process, we will successfully be able to transform San Vicente's daily roles into a more effective process that saves and helps clean up the environment.

1.2 Related Work

Creating an efficient recycling system and tracking process for recyclables is a problem that many countries face. This problem exists in different places where the government or non-profit organizations are involved in the process of material recycling. Several countries have resorted to applications to manage their recycling processes.

Estonia developed Tootjavastutusorganisatsioon (TVO) [4], an application that tracks recycling processes in real-time. The app also provides detailed statistics on different types of recyclables. This application ensures systematic tracking of recyclables, helping Estonia create a cleaner environment.

India developed Swachhata-MoHUA [14], an application that allows users to report waste-related issues that they find in their city. Once reported, the issue is then assigned to a sanitary inspector. This inspector, who is in charge of the specific district in which the issue was reported, will oversee and fix the issue. This application advocates for greater sustainability in India and allows citizens to be more involved in creating a greener community in their city.

Canada developed Recycle BC [1], an application that allows users to view recycling guidelines, create custom recycling schedules, and contribute to a forum to share advice about recycling. This application highlights waste management education and community participation, making recycling more accessible for citizens.

These three examples prove that full-stack applications have succeeded in improving the recycling systems in other countries. In the context of our project, the current system that exists in Montevideo includes a paper and pencil reporting system and a Google Form to track data. This method is not scalable, is hard to maintain, and does not aid collaboration. Therefore, we aim to develop MonteRecycle, a full-stack application and website that will help Uruguay streamline the data collection and reporting process for their recycling system.

1.3 Objectives

MonteRecycle is developed to provide an easier way for the field workers and admins to manage their data for recyclables and their donors. Following are the main objectives of this project:

1. Investigate about the needs of the organization and collect their suggestions for the main functionalities to be supported by the MonteRecycle app.

2. Develop an easy-to-use application for field workers, that can collect and store information with ease. Additionally, the application should provide error warnings (especially for material weight calculation) to reduce human error done by field workers with limited technological and mathematical skills.

3. Develop a web app for administrators to analyze data and generate monthly client reports.

1.4 Our approach

MonteRecycle is unique in its design, due to its full stack structure which seamlessly connects the data-entering user interface used by the field workers to a database hosted on the cloud for data management and persistence purposes. This design will keep all three roles in the organization—the receptor, the classifier, and the admin—connected seamlessly and create a collaborative work environment. Our app and website solution is scalable because it allows for seamless CRUD (Create, Read, Update, Delete) operations in a minimalistic UI and reduces the need for keeping track of numerous papers.

Recycling is a critical component of sustainable waste management. According to the World Bank, approximately 44% of the world's waste could be recycled [17]. However, a significant portion of recyclable materials ends up in landfills due to inefficient sorting and classification processes. In Uruguay, like many other countries, improving the efficiency of recycling systems is essential for environmental sustainability [16].

One of the key issues in recycling is the misclassification of recyclable materials. Studies have shown that up to 25% of recyclable materials are misclassified, which leads to contamination of recycling streams and reduces the overall effectiveness of recycling programs. Contamination can be costly; for example, data from New York City suggests that the average cost of collecting contaminated recyclable loads is significantly higher than non-contaminated loads [2]. Our application addresses this issue by providing a detailed classification system that allows for accurate sorting of materials at the source. This will help San Vicente reduce misclassification and improve the quality of recycled materials.

The importance of efficient and effective recycling systems cannot be overstated. By transitioning from a paper-based to a digital system, San Vicente will be able to track recycling data more accurately and efficiently, ensuring better compliance with waste management standards.

Furthermore, digital platforms facilitate better data analysis and decision-making. According to a report by McKinsey and Company, companies that leverage data analytics in their operations can achieve productivity gains of 6 to 8% [15]. By providing San Vicente with an application that includes in-depth classification, easy data entry for collectors, dynamic material lists, and client location management, we are equipping them with the tools to harness the power of data.

Finally, our approach is not only about functionality but also about user experience. A user-friendly interface ensures that the app can be effectively used by people with varying levels of technical expertise. Research by the Nielsen Norman Group highlights that intuitive design and ease of use are critical factors in the successful adoption of new technologies [18]. By prioritizing these elements, we ensure that the transition from a paper-based system to our digital solution will be smooth and beneficial for all users involved.

By addressing these aspects, our project aims to provide San Vicente with a comprehensive and efficient tool that will transform their recycling management practices, ultimately contributing to a more sustainable environment.

Chapter 2

User Research

2.1 Methods

Since we are making a product that will be used out in the field for San Vicente, we needed to have a consistent meeting time in order to discuss progress, changes, and updates. We first interviewed the representative from the organization and discussed the needs that the organization had and their current methods. Next, we created Figma design pages in order to show our vision for the user interface. We then continued on for several weeks: receiving feedback, applying the feedback, and presenting our revised ideas to the team, until the designs were finalized and approved by the organization.

2.2 Stakeholder needs

The stakeholders and main categories of people we are catering towards are the clients, receptors, classifiers, and administrators at the organization. The priority is ease of use for all involved parties.

- Client: The clients of the organization needed a report of all of their collections. They also would provide information about how the pick up should be done. For example, the exact location of material to be picked up, any time range for pick up, etc.
- Receptor: Receptors needed an easier way to track and store data prom pickups they've made. Our application stores the collected pickup in the database and displays all past collections made by the receptor in the past collections page.
- Classifier: Classifiers needed a way to more easily classify all of their collected materials. Our classify page features easy transition from collection to classification. It also provides colored backgrounds for each of the different materials and the colors are sorted my material type.
- Admin: Administrators oversee all activities within the system. They need comprehensive access to view all pickups and classifications of materials, and the ability to modify any data to maintain the system's integrity and

accuracy. Additionally, they need the capability to create and manage database tables for system organization and reporting purposes.

2.3 User stories

To ensure that our application meets the needs of all its users, we have developed user stories for each role within the system. These stories help us understand the specific requirements and goals of each user type, guiding the development process to create a more user-centered design.

- Client: Clients need to be able to generate reports to analyze trends and make informed decisions about their operations.
  - User Story: ”As a client, I want to generate monthly collection reports to analyze operational trends.”
- Receptor: Receptors are responsible for logging pickups, and it is crucial for them to have a straightforward and efficient way to record data accurately and promptly.
  - User Story:”As a receptor, I want to easily log each pickup using a mobile device to ensure data accuracy and timeliness.”
- Classifier: Classifiers need a system that allows them to quickly and accurately categorize materials to enhance the efficiency of the sorting process.
  - User Story: “As a classifier, I need to quickly categorize collected materials using predefined criteria to improve sorting efficiency.”
- Admin: Administrators oversee all activities within the system, requiring comprehensive access and the ability to modify any data to maintain the system’s integrity and accuracy.
  - User Story: “As an admin, I need to oversee all system activities and be able to edit any data point to maintain system integrity.”

These user stories provide a clear and concise understanding of the different user needs and help ensure that the development of the MonteRecycle app is aligned with the expectations and requirements of its users.

Chapter 3

Design and Rationale

3.1 Designs

Our system's design consists of three main diagrams that illustrate different aspects of the MonteRecycle app. These diagrams provide detailed insights into the functionality, structure, and architecture of the application.

Use Case Diagram:

The MonteRecycle app comprises three primary roles: Receptor, Classifier, and Admin, each connected through various data flows. The Receptor picks up bags and enters their information into the app. The Classifier then categorizes the materials from the bags, and the Admin manages the overall process, ensuring data integrity and generating reports. This process is depicted in Figure 3.1, which shows the different use cases for each role, including logging in, collecting data, classifying materials, and viewing reports. This diagram highlights the interaction between the users and the system, ensuring clarity in understanding the app's functionalities.

Class Diagram:

The class diagram describes the structure of the MonteRecycle system by showing its classes, attributes, operations, and the relationships among objects. Key classes include User, Pickup, and Material, each with specific attributes and methods that define their roles and interactions within the system. For example, the User class has attributes such as name and role, and methods like login and logout. The Pickup class includes attributes like client, location, and weight, and methods for adding and classifying pickups. This diagram provides a detailed view of the system's components and their interactions, ensuring a robust and scalable design.

System Container Diagram:

The system container diagram provides a high-level overview of the system's architecture. It shows how the MonteRecycle app is divided into various containers, including the web application, mobile application, API, and database. The mobile application, built with React Native, includes pages for login, collecting data, classifying materials, and viewing past collections. The web application, built with React, serves as the admin interface for managing users, pickups, and generating reports. The API facilitates communication between the mobile and web applications and the database, ensuring data consistency and integrity. This diagram ensures that all parts of the system work together seamlessly to provide a cohesive user experience.

Entity Relationship Diagram (ERD):

The Entity Relationship Diagram (Figure 3.4) shows the logical structure of the MonteRecycle database. It includes entities such as Location, Client, Pickup, Bag, and Category, each with their respective attributes. The relationships between these entities are also depicted, demonstrating how they interact with one another. For instance, a Client can have multiple Pickups, and each Pickup can include several Bags with specific Colors and Weights. Similarly, each Bag is classified into Categories based on the material types. This diagram is essential for understanding the database schema and ensuring accurate data management.

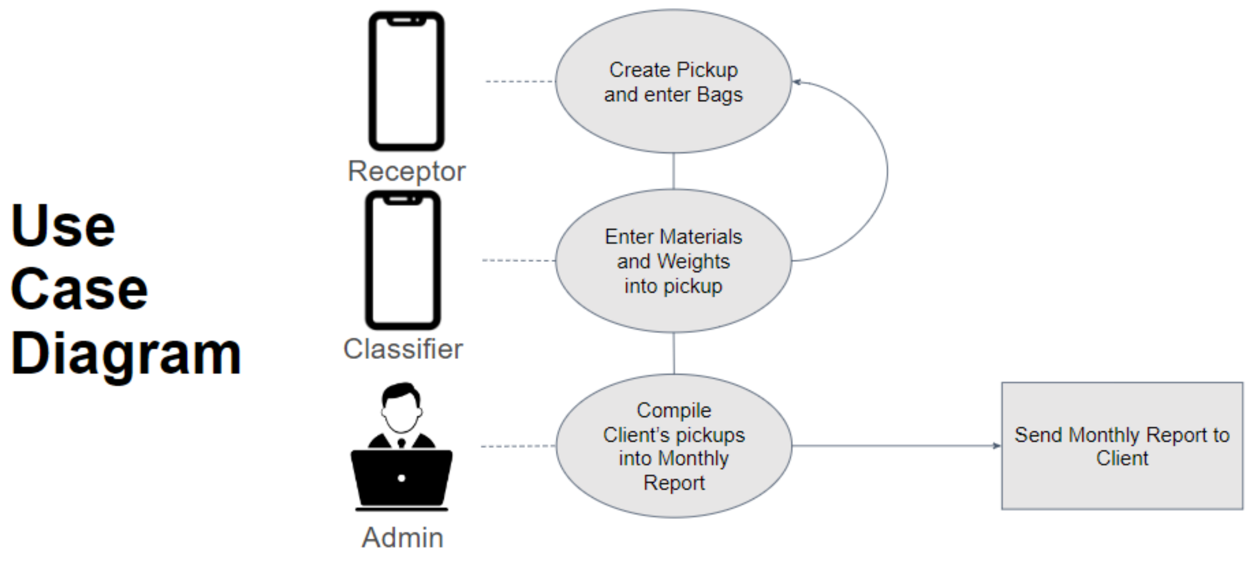
Use Case Diagram
Receptor
Classifier
Admin
Create Pickup and enter Bags
Enter Materials and Weights into pickup
Compile Client's pickups into Monthly Report
Send Monthly Report to Client

Figure 3.1: Use Case Diagram

3.2 Functional requirements

The landing page serves as the initial interface, prompting users to select their login credentials. The home page offers three primary options: collect, classify, and past collections. The collect page enables receptors and collectors to make pickups and subsequently send the saved pickup data to the database. On the classify page, users can split the different materials in a given bag into various sub-classifications of materials. Additionally, the admin page provides functionalities to add, edit, and delete users and material classifications. It also allows viewing all saved tables from the database and sending monthly reports to clients.

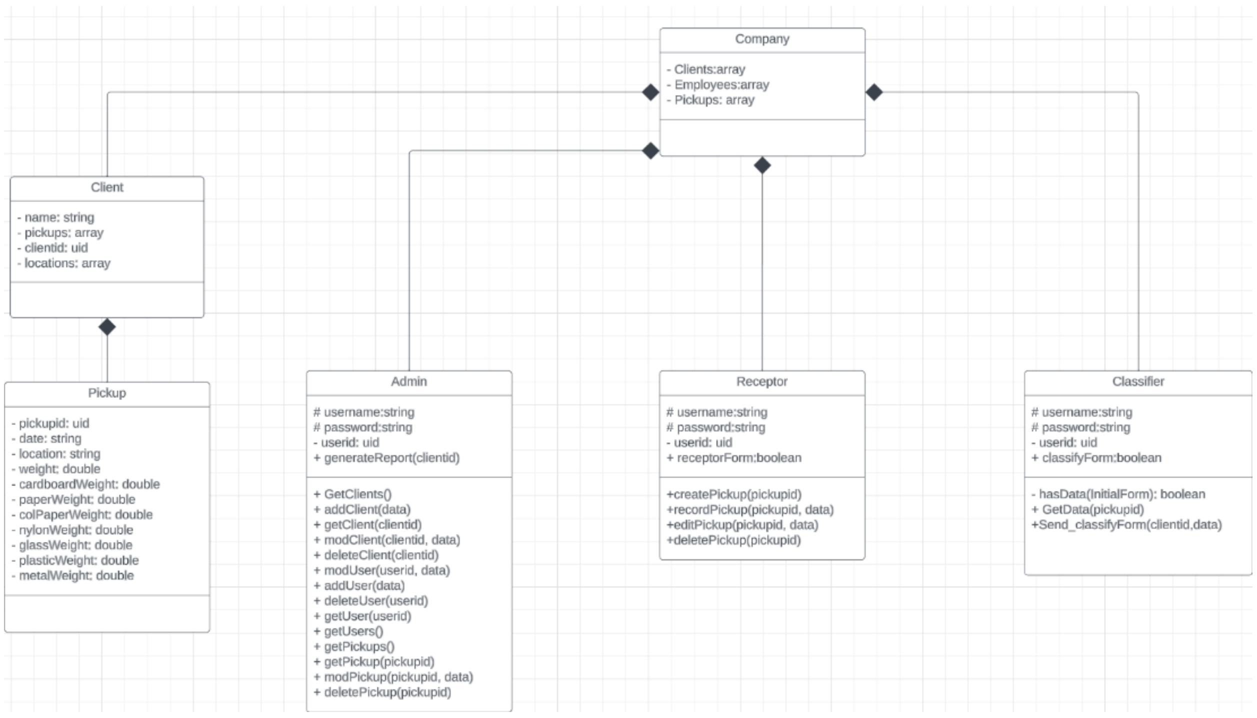

Figure 3.2: Class Diagram

3.3 Non-functional requirements

The non-functional requirements of the application emphasize its performance and usability. It must be fast to load to ensure the best and most efficient experience in the field. The application needs to be easy to use since it will be employed daily by a variety of users in diverse environments. Moreover, robust error handling is crucial; the application must provide feedback when users interact in unexpected ways and offer information on the actions taking place.

3.4 Rationale

Our system is designed based on weekly feedback from the organization contact as well as our advisors. Our application's UI was based on the Figma designs we created for the organization and suggestions from the organization. We wanted the organization to have as much input as possible because they will ultimately be the ones using it on a daily basis. The overall system is designed to have users, clients, and admins. We did it this way based off of feedback and needs of the organization. Since this is a project where communication with the organization was key, we were able to design the system to meet their needs and change certain parts based on their feedback. Due to this, we did not need to explore many other alternatives for system design.

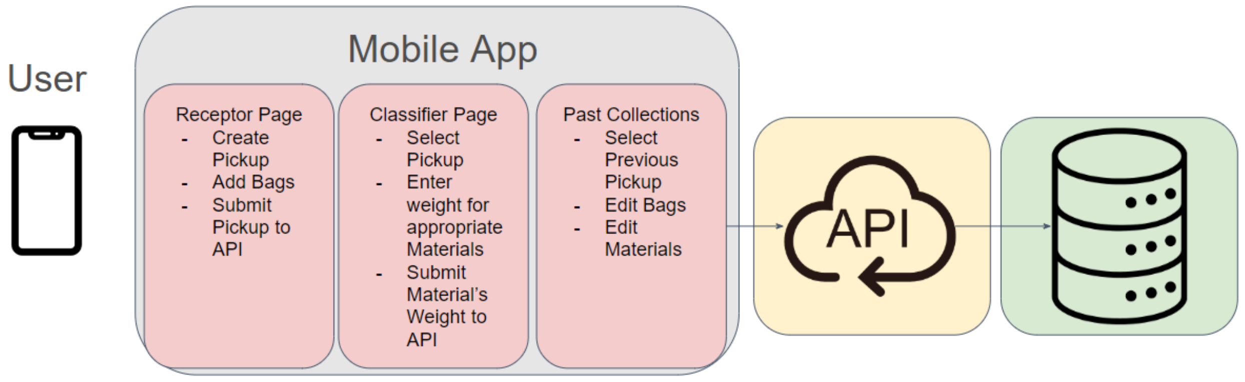

Figure 3.3: System Container Diagram

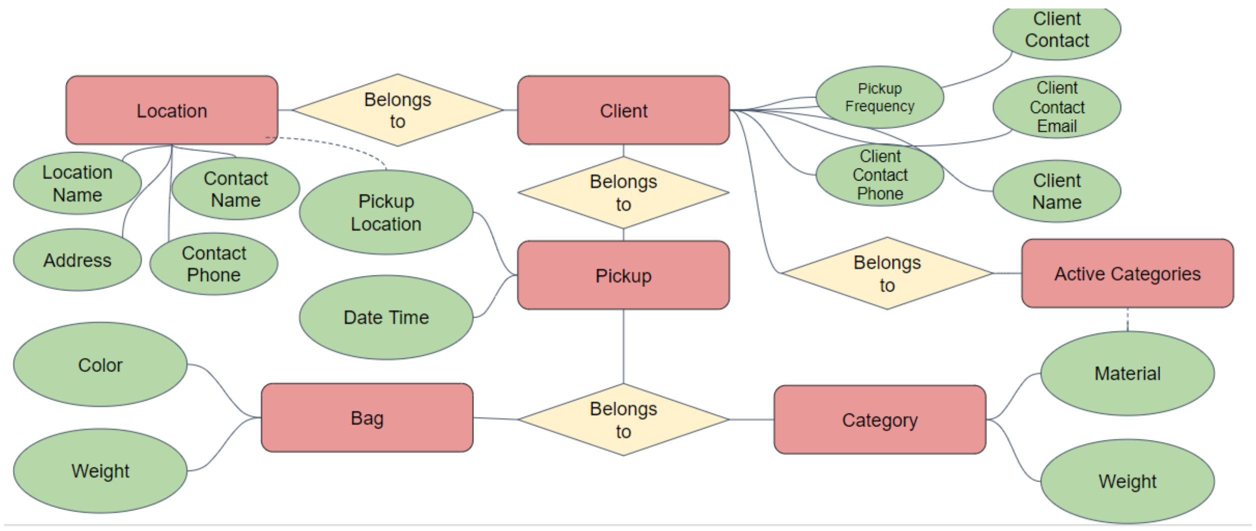

Figure 3.4: Entity Relationship Diagram

Chapter 4

Technologies

The technologies used were all chosen because of the need for a simple yet effective approach to designing the project. The biggest hurdle was planning out the hosting strategy for each piece of the stack. The finished product works extremely well to keep the project under budget and scalable.

4.1 Mobile App

In the end, we delivered a multi-modal full-stack application to the San Vicente organization that allows them to collect data and report it to their clients in a simple yet powerful form. There are four main parts to the finished project: the web application, the mobile application, the API, and the database.

First, the mobile application built with React Native has a login page, a landing page, and three pages in which the user can go in-depth on recycling data. The login page has a list of the employees and allows the user to select their name and enter the landing page. The landing page has three-page options: Collect, Classify, and Past Collections.

The collect page allows the user to select a client, then the client's location, and finally, they can enter a number of bags, what color those bags are, and their weight. Once all data is collected, the user submits the data to the API with a POST request and creates a pickup.

The next page is the classify page, where the user selects a pickup to classify, and as they classify the pickup, they enter a weight to each material category and will add up the entries for the user. Once they have completed classifying the pickup, the user submits the entries using a PATCH request to the API.

Finally, the past collections page is where the user can see all collected and classified pickups and edit all past pickups. The intent of the mobile application is to give the collectors and classifiers a simple app to follow their workflow during the course of a pickup and it is successful at that.

4.2 Web App

The web application built with React and hosted on Google Cloud Platform (GCP), contains a landing page and five pages: clients, client details, user, user details, and materials. The landing page gives the user a choice among three pages to navigate to: clients, user, and materials.

First, the clients page displays a table of all clients, the first column is a hyperlink to the respective client's detail page, the second column is a list of all of the client's locations, the third column is the percentage of the client's total weight that is reusable, and the last column is the days the client has pickups. Also, there is an add client popup on this page that allows the user to enter the client's details; once the user hits submit, it sends a POST request to the API.

Next, the client details page shows the client's name, contact information, location information, percent of total reusable weight, and pickup frequency. There are also buttons to edit and delete the client. The edit button results in a popup that allows editing all information shown in the details portion of the client details page, and sends a PATCH request to the API once complete. The delete button results in a popup asking the user if they want to delete the client and if confirmed, it sends a DELETE request to the API. Below the client details, there is a detailed table of all pickups from the client. The first four columns are set to be the status of the pickup, the date and time, the total weight in kilograms, and the percent reusable. After that, the columns are the different classified materials and their corresponding weights to the pickup. With a large number of materials, one request from the client was to only show materials that have been entered from that client, so this pickup table dynamically creates the material columns so that it is only shown if the type has been picked up from the client.

As well as displaying the pickup data for users, this data all goes into the monthly report function that generates a CSV file with a complete report on the client's pickups in a specified month, and all the months before that within a year. The user can hit an export information button which opens a small popup with a month selection field, then a generate button. Once the file is generated, there is a download report button which starts the download process onto the user's local machine. The intention of this was to give the user the chance to edit the report before sending it to the client, but also easy to send immediately after generating the report.

The second menu options page is the users page. This included a table where the first column is people's names which hyperlink to the person's respective user's detail page and the second is the user's role. There is also an add user button which leads to a popup that asks for the user's first name, last name, and role. Once the user submits the form, it sends a POST request to the API.

After clicking one of the user hyperlinks, it opens up to the user detail page which displays the user's name and their role. There is also an edit button and a delete button. The edit button opens a popup that allows the user to edit the name and role. Once submitted, it sends a PATCH request to the API. The delete button opens a popup asking to confirm deleting the user. Once confirmed, it sends a DELETE request to the API.

Lastly, there is a materials page to dynamically change the materials list. The client asked for this feature with less than a month left in the project, so sadly it wasn't able to be implemented on the mobile app side, however, implemented in the web app and API and is ready to be implemented in the app. This page lists all of the active material options, and an edit and delete button next to each option. If the edit button is selected, it changes the paragraph tag into a text entry box and a submit button below it. Once submitted, it sends a PATCH request to the API. The delete button results in a popup that asks the user to confirm the deletion, and once confirmed, it sends a DELETE request to the API. Lastly, there is an add material function that allows the user to add materials and sends a POST request to the API. Ideally, this would integrate into the classify page, and create a new material category for the classifier to use.

The web app is successful in giving the admin the ability to control the data entry methods for the collectors and classifiers. The app is especially successful in fulfilling the client's request for monthly reports.

4.3 API

The next part of the project's stack is the API; this was developed with Django and hosted on GCP. There are four main URLs to make requests to client, pickup, user, and categories. The client URL has two variations, one with a client ID and one without. There are four views to the client's URL, the GET, POST, PATCH, and DELETE views. The GET view is where the ID variation matters the most and is the most used view. The GET with an ID returns the client's details, their location details, and their pickups. The GET without an ID returns all of the clients, each one including their details and locations. The POST request can only be made without the ID and allows the web app to add a client. This takes in the client's name, the client's contact information, their pickup frequency, and their locations, which contains the location name, address, and location contact. The PATCH view takes in an ID on the URL and can edit all of the information from the POST view. Finally, the DELETE view also takes in an ID on the URL and can delete the row from the API. One important aspect of this is that the DELETE on clients is cascading, so this will delete all of their locations and pickups. While it is risky to set it up like this because two wrong clicks could result in a lot of pickup data being lost, with best practices, there will be a backup of the database. This requires quick action to know a mistake is made. The upside of this effect is that when done correctly, all of the data will be off the database and take up less storage, which goes into the client's request for a low-cost solution.

After the client view, the next most important is the pickup view. This also has four methods: GET, POST, PATCH, and DELETE. The GET view has a similar difference between the ID variation in the URL, when called without the ID, it gives all of the pickups with its details and the client details. When called with the ID, it returns the specific pickup with the details, the client details, and the bags and material categories. The POST request is made from the mobile app without an ID in the URL, and this requires a client ID, a location ID, and the bags to create the pickup.

One thing to note is that the bags can be entered as an empty array, however, it is expected on the POST request because they are created when submitted at the site. The PATCH request takes an ID in the URL and accepts all the same information as the POST, however also expects the material categories to be included in the payload because that is the first time the pickup will be edited and will update the status to ‘C’ to indicate the pickup has been classified. Finally, there is a DELETE view for pickups that takes in an ID in the URL, however unlikely to be used because the client insisted the app should not be allowed to delete pickups.

Next, the user view has the same four request types, but gives much less information and doesn’t change much based on the ID variation. The GET request can be passed with either an ID in the URL or no ID. The request with the ID will return the singular user while the request with no ID will return all users. The POST request is sent with no ID and will include a first name, last name, and role type, which can be admin, collector, classifier, or both. The PATCH request takes in an ID and can edit the same fields as the POST for the respective user. Finally, the DELETE request takes in an ID and will delete the user.

Lastly, the material view has the same four request types but no ID variations. The GET request will return all of the material types. The POST request only needs a name field to create a new material. The PATCH request will take in the ID as input and update the name. Finally, the DELETE request will take in an ID and delete the material.

4.4 Database

The database was made with PostgreSQL which follows the ISO/IEC 9075 standard and ensures proper use of PostgreSQL. This is a well-known relational database with a large community for troubleshooting. Django is very powerful when paired with a relational database due to the ability to migrate the table types and columns and use the models to correctly enter data into the database.

This database was the most important when it came to being cost-effective because the cheapest industry options hosted by GCP or AWS would cost somewhere around \$100/month due to the constant uptime, but no rate increases based on usage. However, many companies know that hobby-level developers are looking for constant uptime but not the same memory and usage constraints that industry options expect. When working with the Frugal Innovation Hub to begin the transfer to them for maintenance, Tanmay Singla suggested Supabase, which has a free tier and forgiving rates once the free tier limits are passed, where it would cost less than a cent a day for the expected usage of the apps. Supabase’s marketed feature is that it can also be used as a PostgreSQL database. The best part of it is that it is open source and only requires payment if the user goes over 500 MB of database space or 5M GB Bandwidth. This proved to be an extremely low-cost option that will help to make the organization even more successful.

Chapter 5

System Evaluation

5.1 Internal Testing

Our testing process is divided into several key stages to ensure the application meets high-quality standards before its release. These stages include unit tests, integration tests, and stress testing within the development team, followed by external testing once the initial quality is confirmed.

Unit Testing:

- Step 1: Begin with the login page. Verify that the list of users is correctly pulled from the database.
- Step 2: Move to the collect page and perform a test collection. Ensure that the collection data is accurately saved.
- Step 3: Navigate to the past collections page. Check if the test collection appears correctly, confirming that the data has been stored.
- Step 4: Go to the classify page for the test collection. Enter classifications for the selected materials.
- Step 5: Submit the classifications. Revisit the past collections page to verify that the collection status has updated from ’P’ (Pending) to ’C’ (Completed).

Integration Testing:

- Step 1: Test the interaction between the login page and the database to ensure seamless user authentication.
- Step 2: Check the flow from the collect page to the database and then to the past collections page, ensuring data consistency and integrity across the application.
- Step 3: Validate the integration between the classify page and the database, confirming that material classifications are correctly recorded and retrieved.

Stress Testing:

- Step 1: Simulate high user traffic and data load on the login page to test the system's performance under stress.
- Step 2: Perform stress tests on the collect page by creating multiple collections simultaneously and verifying the system's ability to handle high volumes of data input.
- Step 3: Apply stress tests to the classify page, ensuring that it can process and save a large number of material classifications without performance degradation.

External Testing:

- Step 1: Once internal testing confirms the application's stability and functionality, release the app to a group of external testers.
- Step 2: Collect feedback from external testers regarding user experience, performance, and any bugs encountered.
- Step 3: Address any issues reported by external testers and perform necessary adjustments to the application.
- Step 4: Conduct a final round of internal tests to ensure all reported issues are resolved and the application is ready for deployment.

By following these structured testing steps, we aim to deliver a robust, reliable, and user-friendly application for San Vicente's recycling management needs.

5.2 External Testing

External early tests with the organization will lead to ironing out all of the major and minor UI/UX issues they may face. With the provided feedback we will be able to accurately assess what issues need to be solved. The user interface needs to be simple and easy to use, due to us working on the application we are less likely to catch minor issues a user with no experience with our app would face. In terms of getting out app to them we have a QR code where the organization can have the phones they will be using scan to download. We have provided video tutorials for them as well to navigate through and how to use the app.

Chapter 6

Implementation Plan

6.1 Timeline

As Figure 6.1 shows, we started our project by conducting interviews with San Vicente. This process roughly took around one month (September 2023 - October 2023), during which we were able to define the components the organization needs in their recycling management system.

Following the interviews, we moved on to the design phase, which spanned from October 2023 to November 2023. During this period, we focused on designing the user interface and defining the features of both the web and mobile applications. This phase was crucial to ensure that the design met the requirements gathered from the initial interviews.

In December 2023, we entered the methodology research phase. This phase involved researching the best practices and methodologies for developing and implementing the application. This included exploring various software development methodologies, data management strategies, and user interface design principles. The goal was to create a solid foundation for the development process.

The development phase began in January 2024 and continued through to April 2024. During these months, we developed the mobile application, web application, API, and database. This phase involved coding, testing, and iterating on the application to ensure it met all functional requirements and provided a seamless user experience.

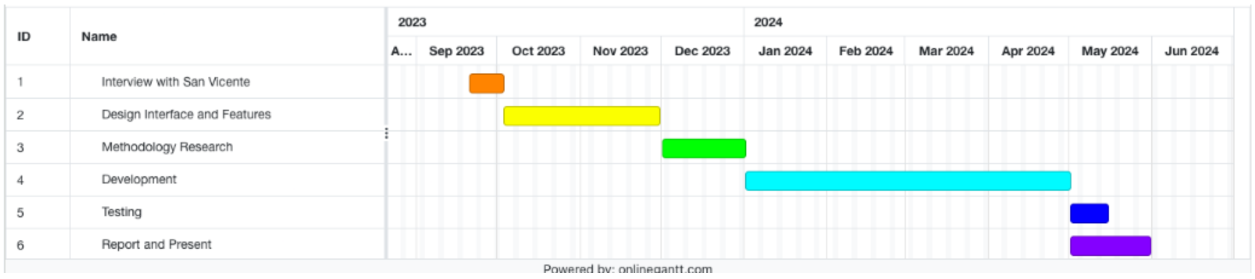

Figure 6.1: Monte Recycle Frameworks Development Timeline

6.2 Agile software development

Practicing agile software development, we had weekly meetings with the representative from San Vicente. These meetings acted as a sprints every week where we gave updates and asked questions. With the work divided amongst the team each of us worked during the sprints to complete the requirements for each of our different types of users. An example sprint for the Classifier page would look as follows: First, meet with the organization and show UI interactions. Next, receive feedback on various changes that need to be made and quality of life improvements. Then, meet again next week with progress on the changes and updates. Lastly, in the following week, present the changes and receive feedback.

6.3 Project Risks

Each project conducted for an external user entails risks that may impact its success. Since MonteRecycle is developed by English-speakers for Spanish-speakers, the first project risk was identified as language barrier. To ensure the swift development of application, we decided to first develop the app in English and then translate it to Spanish. Throughout this process, we were benefited that the San Vicente Organization's representative is a bilingual speaker fluent in both English and Spanish. Privacy and security is a major concern. Therefore, instead of publishing the app, we will send Android APK files in order to only the application installed only on the organization's devices. Because of this we are able to not have a login feature with usernames and passwords for each user because this application will only be used on the phones of the organization and risk of a data breach is lowered. With this method only trusted users of the organization are able to use the application.

Chapter 7

Constraints and Standards

7.1 Constraints

The project faces several key constraints that impact its development and implementation. Firstly, hosting costs need to be kept low due to the significant difference between American hosting options and the organization's revenue. This presents a high risk because the entire project relies on the software generating profits without putting the recycling business in the red. Secondly, the mobile app needs to be simple to use since the users have limited mathematical and technological skills. This is a medium risk as users can be trained to understand the app, but the learning curve needs to be minimized to ensure ease of use. Developing the app for a Spanish-speaking client also poses a high risk. The initial development was done in English, and translating it to Spanish took considerable time. If users cannot understand the app, they will struggle to use it effectively, making proper localization essential.

Another constraint is the time difference between the client and developers. Weekly meetings with the client were challenging due to a 4-5 hour time difference and changing class schedules. Although this is a low risk, it was a hurdle to communicate progress efficiently, though alternative methods were used to keep the client informed. The large developer and organization team size also created a constraint, as connecting various parts developed individually took time and required adapting to changing requirements. This was a medium risk because frequent changes and commitments necessitated significant integration efforts across different projects.

Lastly, the mobile app needs to work well on Android devices, as the majority of phones in Uruguay are Androids. This presents a high risk because if the app does not function properly on the intended devices, a substantial part of the project would be compromised. To address this, React Native, an open-source UI framework , was chosen to ensure compatibility with both iOS and Android devices. The development was tested on both Android and iOS emulators. This approach ensured the app worked seamlessly on both platforms, catering to the prevalent Android users while also supporting iOS for potential future users.

Standard	Component	Description
ISO/IEC 22275	Frontend	Standard ensuring the proper use of JavaScript
ECMA 262	JavaScript ES6	This ensures the proper use of modern Javascript, based around modules type, such as in React and React Native
ISO/IEC 9075	Backend	Standard ensuring the proper use of PostgreSQL
ISO/IEC TR 14759	Mock Up and Prototype	This standard was used during the design phase of the project with Figma
ISO/IEC 19501	Design Phase	UML Diagram standard
ISO 29119-1	Testing	Specifies test processes that can be used to manage and implement software testing for any project
ISO/IEC 12207	Software Life Cycle	Ensures proper development methods
ISO 8601	Date and Time Format	Standard to ensure all can understand date and time

Table 1: Standards and their usage

7.2 Standards

For people outside the team to understand the process and finished product of our project, we used ISO/IEC standards. These standards can be seen in Table 1.

Starting with ISO/IEC 22275, this was the first standard set when beginning the project [11] . It was obvious that React and React Native would be developed with JavaScript, so this one was easy to implement from the beginning.

ECMA 262 is the ECMA script's latest version of the standards around Javascript, most importantly including ES6 which is the version of Javascript used by modern React and React Native apps today [3].

Next, ISO/IEC 9075 helped with backend development. When deciding what database framework to use, knowing that PostgreSQL was already an industry standard and had the relational aspect needed, it became a clear choice [13]. In development, using this standard was easy due to Django's strict relational objects.

When prototyping the designs, the team used Figma to create wireframes which falls under ISO/IEC TR 14759, the standard for Mock Up and Prototype [8].

During the design phase, ISO/IEC 19501 helped convey the system designs and use cases of the project [9]. UML is a strict designing language that helps others understand the technology and how different pieces of the tech stack work together.

At the end of the project, using ISO 29119-1 was helpful in the testing phase of the project [12]. The unit testing was an extremely necessary part of the API and this follows this standard.

Throughout the project, following the software life cycle was always the intended process. Following ISO/IEC

12207, the team stuck to the steps however only got to the testing and integration step and missed the maintenance step [10]. However, as the team hands over the project to the Frugal Innovation Hub, testing and implementation will be completed, and set up the new team for proper maintenance. This was the hardest standard to adhere to due to the limited time of the project and the maintenance step was never truly possible.

Lastly, the product follows ISO 8601 which dictates how date and time is stored [7]. In the frontend, it was easy to implement this because Javascript has a lot of built in functionality to handle ISO date time formatting. These built in libraries made it extremely simple to follow that and work with the date time formatting to make it more human readable when needed.

Chapter 8

Societal Issues

8.1 Ethical

When working with the San Vicente Organization, it was easy to know the product would be used in an ethical way. From an outsider's view, it is a non-profit organization helping clean up Montevideo, Uruguay, and the money they make from the recycling business goes right back into the community. On top of that, they help with education in the area, and the classifiers and collectors are people in the community without formal work experience and this is a chance for them to start a resume. In an ethical sense, the product would be used to clean up the earth and advance people's careers who need it, which is a glowing example of the right thing to do.

While developing the project, the biggest ethical concern had to do with storing personal data. As always, any data stored could be stolen and held hostage. There is a paradigm that the information has to be valuable enough to pay the ransom back for it, but when developing and collecting data, it's a good rule of thumb to act as if the personal is the most valuable part of the project. To help with this, the apps specifically don't collect very sensitive data. For the employees, it is only their first and last name. This is extremely bare in terms of what other companies collect, so if there were a breach, this would have very little harm. When collecting the client's information, there is more data, however still very minimal. When creating a client, it asks for the contacts name, email, and phone number, as well as location contacts name and phone number. While this is a lot more and a targeted breach could reveal their personal information, this is all business contact information that could be obtained through open source intelligence or from the company. The ethical context of the data collection has been mitigated through not collecting sensitive information.

8.2 Social

The social group the project was developed for is only the San Vicente Organization as they are the sole client. That means the only users will be the collectors, classifiers and the admins at the organization. However, this product will have an effect on the larger community in Montevideo through the good work that the organization does to help clean up the area, educate people, and employ people. In terms of unintended consequences, the apps could build a reliance on the project from the organization and if it were to go down or have issues, it could negatively impact the organization, and if it's a big enough impact, possibly affect the whole community. The Frugal Innovation Hub helping maintain the apps will help mitigate that possibility.

8.3 Political

Overall, the greater effect on society will be minimal due to the limited user group of the project and the exclusive nature of the apps. There will not be any random users going to Montevideo and helping sorting recyclable materials, but hopefully the organization will be able to spread more of a butterfly effect, giving people the opportunity to get better jobs and educating more people, and that compounding in the community and bringing more good. Overall, it's difficult to understand the political climate of Uruguay and didn't come into much discussion with the client, but as long as the San Vicente Organization is operating, the project will have the opportunity to do more good in the community.

8.4 Economic

Economic concerns were the biggest priority to the team while developing. If the product costs more to operate in server costs than the value it brings in, what's the point of the project? Wherever possible, cutting the operating costs was considered. Luckily, when developing, this also came as a first thought and the solutions found in the development process worked for the most part when trying to plan out implementation.

First, the mobile app was the easiest to figure out how to host because those all run natively on the device and only need to make calls to the API. With no need for finding how to host it, the only question was how to deploy. Ultimately, an Android Package Kit (APK) became the best solution because it will make it easy to keep the app to a closed group of users. Another solution would be to put it on the Google Play store, however there needs to be a system to restrict users who should not have access to the app.

When it comes to hosting the web app and the API, Google Cloud Platform (GCP) and their Cloud Run service was a clear choice due to how easy deployment was and the low costs that scale linearly. According to the GCP pricing page, the official Cloud Run pricing has a free "first 240,000 vCPU-seconds free per month," and free "first 450,000 GiB-seconds free per month". On top of the free tier pricing, the paid pricing is forgiving, with "$0.40 / million requests beyond free tier" [5]. The organization should never go above the free tier, and if they do, the ten daily users would hardly get close to a million requests in a month.

The biggest challenge was finding a database host that would be cost effective. The issue with a provider like Google Cloud Platform is there isn't a linear cost for scaling like there is for the Cloud Run service. The average cost for the lowest cost database server from Google Cloud would be roughly $26.44 USD per month [6]. While it's not crazy, there are better options. Finally, the decision to use Supabase, an open source Postgres server was the best choice. It has similar free tier options and usage scaling to the GCP Cloud Run which makes it the cheapest option. Again, the expected limited usage will make the operating costs close to zero. The low cost was a major requirement for the project to begin with so the economics side of it was imperative.

8.5 Health and Safety

In reality, the product isn't developed for the public, just a private group of employees from the San Vicente Organization, so the public's health and safety isn't a big concern. At the same time, it does make sense to make sure there is no unsafe part of the product. The intended use is for on job sites so the dangers around them while they're distracted is a concern. At the same time, the job function is not changing the health and safety aspect because the previous solution of writing on a piece of paper functions very similarly to entering the information on the phone. The distraction aspect of the product is the same level the collectors and classifiers have dealt with before the product is implemented.

8.6 Manufacturability

The manufacturability is not much of a concern either because the product is purely software. There is nothing to make or manufacture in the normal sense. The only aspect that may apply to the project is the manufacturability of the devices it runs on. With the mobile app, the decision to develop with React Native allows both Android and iOS devices to run it and those are the only two main operating systems in the world, so any phone anywhere can run it. As well as the website, any computer or device with connection to the internet can connect.

8.7 Sustainability

When it comes to the sustainability of the product, the apps can run as long as the servers are up and the database has storage. Of course, if the current plans are maxed out and the next tier isn't purchased, the apps would fail. That is an unlikely scenario though, the sustainability of software is the same level as the technology it's developed to run on. Of course maintenance is an expected part of it and without proper maintenance, it could fail. The Frugal Innovation Hub is committed to staying on the project to make sure that any technological needs the product has are met. So in a sense, the product has long term sustainability.

8.8 Environmental Impact

When considering the overall impact on the environment, the product will do more good than it could ever do with harm. The recycling forward nature of the client makes the project focus on how to better the environment. Having such a traditionally good motive for making the product made motivation for developing the product easy. While the possible good it does is endless, there will always be the other side of it. When operating servers, there is a energy need from the server farm and by giving Google Cloud and Supabase the business, it's giving into that energy need. The best case scenario is the server farms are using renewable energy to keep the server online, and the current build of the API and web app are on a low carbon regional signal server farm which helps cut down on the possible harm. Regardless, the negative environmental impact of the servers is much less than the possible good the app provides.

On top of that, there is the environmental harm that comes with today's devices and machines. The rare minerals that go into making every processing chip, computer and phone have a negative impact on the environment when mined, but that is a necessary evil when trying to mitigate environmental impact.

8.9 Usability

In the development stage, usability was a major factor in the early designs of the apps. Knowing the collectors and classifiers would have difficulty using technology and limited mathematical skills, a lot of thinking went into the design and how to make the navigation and flow of the mobile app easy to use and understand. It was tailored around the process the user already completes every step of the pickup process. The mobile app does a lot of the adding for the user and even has color coded material categories matching the color of the bag they should come out of. While the mobile app's design is made to be intuitive, there is an expectation that not everyone will understand it at first. This is where our organization contact will be of help as a resource for everyone to fill in the gaps.

8.10 Lifelong learning

Throughout the development process, there were plenty of new technologies that the team learned to complete the final product. Both of the members on the mobile app team were using React Native for the first time, a member of the web app team was using React.js for the first time, and a member was developing an API and database for the first time. The amount of growth and learning across the team was unbelievable and it will absolutely allow everyone to use these new skills in the future. Not only did the team learn the specific skills, learning how to learn new technologies alone through the internet is a skill in itself and the most important in technology today.

8.11 Compassion

Compassion is an extremely important part of this project, whether it be compassion for the San Vicente workers or just compassion for the environment. The nature of the project had compassion in spades and that's mostly due to working with an extremely compassionate organization as a client. The development team truly hopes this helps the organization grow the recycling impact on the Montevideo community and allow them to do even more good, educate more people, and give more people jobs. The engineering with a mission part of the project is so special and something that made all the work worth the effort. While the team will likely not experience the impact first hand, knowing it can only do good in the world is rewarding in itself.

Chapter 9

Conclusion

A part of the software life cycle is completing retrospectives regarding the work just done. This is a necessary part of the life cycle as it gives the team feedback on what was done well and what could be improved. It requires an honest look at the finished product and can often be difficult to look for the flaws in what someone thinks might be perfect. This chapter serves as a great retrospective, highlighting the lessons learned, what went well and what did not, and what to do in the future.

9.1 Conclusions Drawn

Through the process, the team learned a lot about the software development life cycle, specifically working with a client. Some of the specific lessons learned are:

- Pivoting is necessary in the development process
- Clients will want to change designs and the product during development
- The more eyes on the product during the process means more input and options
- User feedback is the most important feedback

Looking back at our project objectives, the project hits the mark on many of them. Firstly, to make the project work, pivoting helped a lot because there were easier solutions out there, and instead of continuing to make the difficult method work, the changed one is going to free members up to work on other parts of the project. For example, the API was originally going to be made with Node.js and express, but the structural needs of PostgreSQL were more important, and Django's framework is set up much better to work with a relational database. This pivot allowed for the request bodies to be worked on with more focus and allowed unit testing to be completed.

Next, working with clients is much different than expected, especially the amount of feedback on designs. While one person may design it one way, the client will always have more knowledge of how the process should work and it is up to the team to figure out how to best make those changes, and then make them. The unexpected part of this is making the changes takes time away from the end goal. Knowing how much change would be made throughout the process, there could be a benefit from longer term work periods with larger meetings with the client, however having their input throughout the process helped shape the apps into what they are now.

Similarly, the more eyes on the project led to finding more spots to tune up the flow of the apps and places it was confusing to new eyes. With the addition of our advisors from the university, we had three sets of fresh eyes on the apps which helped raise questions on why certain things are certain ways, and really got to the root of how it should function. When the developer never gets new eyes on the project, parts that they thought were perfectly fine will not make any sense to a lot of other people.

In regards to that as well, the intended users are going to be the most important feedback. In March, we sent demo videos of the app's flow to the organization, to show the people who will be working with the apps and ask what they thought of the process and the app flows. This brought us the answers to things that we had differing opinions on, and that feedback was extremely valuable to help us finalize it. While there can't be a large pool of opinions on the entire process, giving the users the chance to make adjustments at some point is extremely valuable.

9.2 Advantages and Disadvantages

Throughout the process, there were plenty of successes, however at the same time, a lot of setbacks and hurdles to get across. A few of those disadvantages were issues understanding the recycling process for the organization in the design process, there were a lot of setbacks trying to connect everyone's work when developing, and there was a lot of learning the technologies along the way.

The advantage of the project is that it solves the main problem at hand, making a simple solution for the client to collect, manage, and report their data. The user experience was the most important part because the normal assumptions of the user's capabilities couldn't be made. One big point from the client at the beginning was that the users aren't as mathematically or technologically skilled as the American user set the team has been used to developing for. With this constraint in mind, the design process was crucial to make sure that the users could understand the apps, especially the mobile app where there would be the biggest technological divide. Essentially, it was on the team to design something intuitive for anyone with some knowledge of the recycling pickup process.

The next big success with the project was keeping costs down for the organization when the product was rolled out for everyday use. Understanding the limited size of the user base made a lot of the normal solutions untenable, especially when it came to databases. These expect a lot more storage and bandwidth than what this project needed, so it came to finding solutions intended for hobby projects. These database hosting options expect low storage and bandwidth but just need to be accessible. This project is similar, where there is no expectation of hundreds or thousands of users and looking closer to around tens.

When it comes to disadvantages, there are a good number of them. Firstly, there could have been much more testing done. In the end, the API has unit tests for every view and method which is certainly a start to make sure the product works as intended. However, it needed a full round of user testing to find the spots it could improve. On top of that, some feature testing with the app to make sure what it is sending is exactly what was expected. To do this, it would take an extra view that returns the created row to the app, and the app certifies that the returned object matches what it just sent. While the code is written to work how one would expect, not testing could leave a major flaw and for full completion, its behavior needs to be certain.

Another place the project could be improved is through full documentation of the API. In the time of the handoff between the team and the Frugal Innovation Hub, there will be documentation written, however, formal documentation would be helpful for any frontend developers trying to get the request payload right.

Finally, there is an issue with the security of the apps, however, the current state was a requirement from the client. There was a big fear that the users may forget passwords and there is no way to recover them. The team was asked to make the apps without the normal username and password systems which led to the intended development of it with little security protocol. If given more time on the project, there must have been some solution to this problem, however, as it stands, there are holes in the application. The one upside of this problem is that the apps can be held exclusive to the public and for only the organization's use, so as long as there is no breach on their end, no one would be able to access the apps.

9.3 Future Work

When it comes to the future work of the project, there are some short-term parts to be completed and some features that would take much longer to implement but be extremely valuable. When it comes to the short term, creating a security solution to page log-in would be a quicker fix as well as implementing the dynamic material list on the mobile app. For long-term projects, there is a reselling element to the business that the organization could use tracking for the selling prices of different materials to different buyers. Another piece that would be valuable is a data analysis section on the web app to help understand what most of the reusable materials are and where the clients could improve on their separation efforts.

Starting with the security solution, the most obvious solution would be an OAuth route to allow the user to just log in to a Google account set up on the company mobile devices, and there would be much fewer issues with forgetting passwords. This would of course rely on the linked OAuth account being on the device, however, this problem shouldn't come up too often. It is worth testing and seeing if the users are capable of that, but not a guaranteed solution. Implementing this on the back end wouldn't be too difficult, and just result in using Django's built-in user authentication system instead. There could also be a solution involving biometrics as it has become more popular. The only issue with this solution is the company giving out phones to the classifiers and collectors means that one person may not have the same device as the day before and it may not have their face id or fingerprint saved.

The next short-term problem is the dynamic materials list. This would be a quick solution on the mobile app's end, where it just needs to create a component type of the many material categories that already exist, make a GET request for the material list, and dynamically display that component throughout the screen. There could be improvements to the backend side of it too, by making material categories depending on the color bag it would be found in. However, the system is on the brink of being complete and the team just ran out of time to get other parts functional.

As for the long-term projects, there is a part of the business that was mentioned in the original research process where the sorted materials are then sold to buyers in the area for prices that fluctuate. This part of the business makes a stock market type of aspect which would be exciting to develop and would greatly benefit the organization. While it is a non-profit, the client wants to maximize the money generated from this part of the business so they can make a greater impact on other parts. The general idea of this would be a page that tracks all the buyers in the area, and when they sell a load of reusable material, they log it, and then that can lead to charts that show the trends of each material. From there, the possible technology to maximize this is endless and many fintech principles could be applied to show a huge increase in profit from selling the material. There is huge potential that an AI bot could be implemented to help in this process too.

The other long-term project would be data analysis on specific clients and their pickups. This would be an easier project to implement because the logging logic is already there, it just needs to be compiled and analyzed. This could result in helping clients sort their recyclables better and help the sorting process for the classifiers. This could also go a long way in terms of helping the environment, showing what the biggest polluters they collect are, and possibly suggesting alternatives to them and making Montevideo cleaner on first use. The data analysis aspect of this could result in endless lessons to help the organization run smoother and make the area cleaner.

Regardless, of the short-term issues or long-term dreams, the project needs to get into the user's hands and the system needs to be fully tested, but once that's complete, the project can be rolled out for everyday use and satisfy the project requirements set by the client.

Chapter 10

Acknowledgments

Special thanks to Dr. Jahangiri (Senior Design Project and Thesis advisor), Allan Báez Morales (Director of Frugal Innovation Hub), and Ma.Laura Bergeret (San Vicente's representative) for guiding the team and the project through every step of the way.

Chapter 11

References

[1] Download the app - Recycle BC. URL https://recyclebc.ca/learn/download-the-app/.

[2] Christian Blanco, Calvin Spanbauer, Sara Stienecker. America's broken recycling system. URL https://cmr.berkeley.edu/2023/05/america-s-broken-recycling-system/.

[3] ECMAScript. Ecme-262. URL https://ecma-international.org/wp-content/uploads/ECMA-262_14th_edition_june_2023.pdf.

[4] European Environment Agency. Early warning assessment related to the 2025 targets for municipal waste and packaging waste. URL "https://www.eea.europa.eu/publications/many-eu-member-states/estonia".

[5] Google Cloud. Pricing — cloud run, . URL https://cloud.google.com/run/pricing.

[6] Google Cloud. Pricing — cloud sql for postgresql, . URL https://cloud.google.com/sql/docs/postgres/pricing.

[7] International Organization for Standardization. Iso 8601, 2018. URL https://www.iso.org/iso-8601-date-and-time-format.html.

[8] International Organization for Standardization / International Electrotechnical Commission. Iso/iec tr 14759:1999. URL https://www.iso.org/obp/ui/#iso:std:iso-iec:tr:14759:ed-1:v1:en.

[9] International Organization for Standardization / International Electrotechnical Commission. Iso 19501:2005, 2005. URL https://www.iso.org/standard/32620.html.

[10] International Organization for Standardization / International Electrotechnical Commission. Iso 12207:2017, 2017. URL https://www.iso.org/standard/63712.html.

[11] International Organization for Standardization / International Electrotechnical Commission. Iso 22275:2018, 2018. URL https://www.iso.org/standard/73002.html.

[12] International Organization for Standardization / International Electrotechnical Commission. Iso 29119-1:2022, 2022. URL https://www.iso.org/standard/81291.html.

[13] International Organization for Standardization / International Electrotechnical Commission. Iso/iec 9075-1:2023, 2023. URL https://www.iso.org/standard/76583.html.

[14] Janaagraha. Swachhata-MoHUA on the App Store. URL https://apps.apple.com/us/app/swachhata-mohua/id1124033628.

[15] Nord Sense. The ultimate guide to smart waste management. URL https://nordsense.com/the-ultimate-guide-to-smart-waste-management/.

[16] Scott Mouw. State of curbside recycling report. URL https://recyclingpartnership.org/wp-content/uploads/dlm_uploads/2020/02/2020-State-of-Curbside-Recycling.pdf.

[17] Silpa Kaza, Lisa C. Yao, Perinaz Bhada-Tata, Frank Van-Woerden. What a waste 2.0: A global snapshot of solid waste management to 2050. URL https://openknowledge.worldbank.org/entities/publication/d3f9d45e-115f-559b-b14f-28552410e90a.

[18] United States Environmental Protection Agency. What is integrated solid waste management? URL https://nepis.epa.gov/Exe/ZyPDF.cgi/P1000L3W.PDF?Dockey=P1000L3W.PDF#:~:text=.