The City of SAN DIEGO

Analysis & Design Document

Route Assignment Finder Application

Tracy Pham, Tegan Goodman

CITY OF SAN DIEGO – 8353 Miramar Place, San Diego, CA

1. PRELIMINARY ANALYSIS

1.1 OVERVIEW

The Route Assignment Finder is a Streamlit-based web application designed to help users select, analyze, and visualize waste collection routes for specific dates. It integrates geospatial analysis, database querying, and interactive mapping features to enhance route management and vehicle tracking.

BACKGROUND INFORMATION

The Environmental Services Department (ESD) is responsible for managing waste collection services in San Diego, including refuse, recycling, and green waste pickup. To improve efficiency and sustainability, the department actively tracks collection routes and vehicle movements using GIS data and GPS technology. ESD records waste tonnage data through two key systems: ARTS, where supervisors manually log load data, and RAD, which collects weight measurements from landfill scales.

As part of its ongoing efforts to enhance waste management, ESD is implementing route rebalancing under Measure B, a policy change that amends the San Diego Municipal Code to allow the City to charge fees for solid waste services. This transition requires adjustments to collection routes as some properties shift to private service providers. The goal of rebalancing is to optimize routes for safer, more efficient pickups, reducing operational costs, fuel consumption, and vehicle wear while ensuring a balanced workload among drivers. To support this effort, ESD has partnered with Routeware, a company specializing in route optimization technology.

To maintain data integrity during this transition, ESD developed an app that enables data analysts to monitor collection routes and validate information across multiple systems. The app provides a centralized resource for tracking key metrics, including route paths, driver assignments, truck usage, tonnage collected, and route completion times. By consolidating this information, the tool helps ensure accurate data as the department adjusts and refines its operations under Measure B.

Beyond data validation, the app also serves as a management tool for monitoring driver performance. By analyzing time spent on routes, supervisors can identify inefficiencies, address potential challenges, and provide support to drivers who may need additional guidance. This data-driven approach enhances operational oversight, enabling ESD to make informed decisions that improve overall service quality. Ultimately, the app plays a critical role in facilitating a smooth transition during the rebalancing process, promoting greater efficiency, accountability, and sustainability in waste collection services.

1.2 SCOPE

IN SCOPE

- Developing a web-based application with an interactive user interface.
- Implementing algorithms for route optimization.
- Ensuring secure user authentication and data protection.
- Providing visualization tools for better route management.
- Allowing users to manually override automated recommendations.
- Generating reports based on route assignments and optimizations.

OUT OF SCOPE

- Integration with third-party mapping services beyond basic visualization.
- Real-time GPS tracking and live route updates.
- Mobile application development (the initial version is web-based only).
- Advanced AI-based predictive analytics beyond rule-based optimizations.
- Handling logistics and fleet management beyond route assignments.

1.3 ASSUMPTIONS

- The Collection Services Division of the Environmental Services Department of the City of San Diego will provide all operational data, including vehicle data, route information, and any necessary logistical details.
- All spatial data will be regularly updated, accurate, and accessible through reliable systems, ensuring that users have access to the most current data available.
- The application assumes users have a basic understanding of operation and functionality of CSD, including selecting routes and vehicles for analysis.
- All analysis and decision-making will be based on data collected from GPS tracking on trucks and from landfill operations.
- Users must download and install the necessary files or software components required to operate the application, ensuring that the system is fully functional upon use.
- A stable and consistent Wi-Fi connection is required for data retrieval, processing, and communication with the application.

- The software will be compatible with commonly used operating systems, including Windows, macOS, iOS, and Android, and will work across devices without requiring specific hardware configurations.
- Users are expected to have access to and be familiar with the necessary hardware (e.g., computers, tablets, or smartphones) to run the application efficiently.

1.4 CONSTRAINTS

- The application must operate within the defined date range of August 2024, restricting its functionality to these months for data input, analysis, and reporting.
- The application relies on data from ARTS/RAD, so its accuracy and performance are dependent on the integrity and quality of the data provided by these systems. Any inconsistencies or errors in the source data may affect the application's functionality and outputs.
- Staff availability may impact the project timeline, causing delays in development, testing, deployment, and support.
- The application is locally hosted, meaning it will be managed on internal servers, limiting scalability and flexibility compared to a cloud-based solution.
- The application must comply with the city’s internal data security and privacy policies, which may impose limitations on data sharing, access control, and user authentication.
- User access to the application may be limited by internal IT policies, including permissions, security protocols, or access to specific network resources.
- System performance may be constrained by the local infrastructure, including the local user’s PC... server capacity or hardware restrictions within the hosting environment.

1.5 DEPENDENCIES

- GIS data, including routes, vehicles, and landfill info, must be regularly updated to ensure accurate visualizations and reports.
- The application relies on DuckDB for data storage and querying. Any database issues, like downtime or maintenance, could affect functionality and data access.
- User access depends on compatible browsers, network settings, and IT resources. Changes in IT policies could affect access or functionality.

2. FEATURES

2.1 USER INTERFACE

1. **Visual Branding**
    a. Displays images, logos, and other visuals for branding.
2. **Layout and Input Fields**
    a. Clean, user-friendly layout.
    b. Input fields for user interactions, with specific areas for:
        i. **Waste commodity selection** (dropdown or selection options).
        ii. **Route number input** (5-digit route number field).
        iii. **Date Picker**: Restricts user to selecting dates only in August and September 2024.
3. **Data Visualization**
    a. Clear display of relevant visualizations such as:
        i. **Vehicle counts** for selected routes.
        ii. **Route maps** based on selected commodity types.
        iii. **Route-specific data** such as missed pickups, population, census data

For details on the user interface, refer to Appendix 5.2.

2.2 BACKEND

1. **Database Connection**
    a. Utilizes DuckDB to connect to and query the data.
    b. Loads spatial extension for processing geospatial data.
2. **Data Querying and Processing**
    a. Queries vehicle data from GPS tracking tables.
    b. Filters data based on:
        i. Selected route
        ii. Date range (August and September 2024)
        iii. Vehicle number (5-digit route number).
    c. Loads shapefiles for route layers, specifically tailored to each commodity type.
3. **Vehicle Tracking and Analysis**
    a. Queries RAD and ARTS databases for load and material details.
    b. Displays top vehicle counts for selected routes.
    c. Allows for vehicle selection based on available data.
4. **Geospatial Processing**
    a. Converts GPS data into GeoDataFrame format for spatial analysis.
    b. Performs spatial joins with route and landfill layers.
    c. Identifies landfill visit times and organizes route tracking:
        i. Before and after landfill visits.

5. **Additional Information Display**
    a. Fetches route-specific information such as:
        i. **Missed pickups**.
        ii. **Population data**.
        iii. **Census data**.
    b. Conducts spatial joins to assess landfill interactions with routes.
6. **Error Handling**
    a. Implements exception handling for potential issues like:
        i. Missing data.
        ii. Index errors.
    b. Ensures robust processing of timestamped route activities to avoid data inconsistencies.

3. TECHNOLOGIES USED

3.1 INTEGRATED DEVELOPMENT ENVIRONMENT (IDE)

An Integrated Development Environment (IDE) is a software application that provides a comprehensive workspace for developers to write, test, and debug code efficiently. IDEs integrate multiple development tools into a single interface, including a code editor, compiler or interpreter, debugger, and project management tools. This setup allows developers to streamline their workflow, reducing context switching and improving productivity.

Key features of an IDE include:

- **Syntax highlighting** – Enhances readability and helps identify errors.
- **Code completion** – Suggests function names, variables, and syntax elements to speed up coding.
- **Integrated debugging** – Provides tools to identify and fix errors in real-time.
- **Version control support** – Allows seamless integration with Git and other version control systems for tracking changes.
- **Project navigation** – Enables easy management of files and dependencies within a structured workspace.

For our specific use case, an IDE is essential for developing and maintaining our Streamlit application. It enables efficient debugging, testing, and real-time modification of our Python scripts, ensuring a smooth development process. Given our focus on geospatial analysis and data visualization, an IDE such as Visual Studio Code (VS Code) or PyCharm provides powerful tools to write Python scripts, test GIS libraries like geopandas, and integrate with databases like DuckDB. The use of an IDE allows us to efficiently iterate on our app, add features, and ensure a stable and scalable application.

3.2 PYTHON ENVIRONMENT

A Python environment refers to the setup that includes the Python interpreter, libraries, and dependencies required to run Python code. Managing Python environments is crucial for maintaining consistency across projects and avoiding dependency conflicts. There are several types of Python environments:

- **System-wide Python Installation** – Installed globally and used across different projects, but can lead to version conflicts.
- **Virtual Environments (venv or virtualenv)** – Isolated environments for specific projects, ensuring dependency control.

- **Conda Environments** – Used in data science workflows, allowing package management for scientific computing.
- **Docker/Containerized Environments** – Provide reproducibility by encapsulating code, dependencies, and system configurations in a container.

For our use case, we need a well-structured Python environment to run our Streamlit application and handle geospatial data processing. We will likely use a virtual environment (venv or conda) to manage dependencies, ensuring compatibility with libraries such as pandas, geopandas, streamlit, folium, and duckdb. This setup allows us to keep our development workspace organized and prevents version mismatches between different projects. Additionally, cloud-based environments like Google Colab or Jupyter Notebooks can be useful for preliminary data exploration before integrating findings into our main application.

3.3 GOOGLE EARTH ENGINE (GEE)

Google Earth Engine (GEE) is a cloud-based geospatial analysis platform designed for large-scale environmental monitoring, remote sensing, and spatial data analysis. It provides access to a vast repository of satellite imagery and geospatial datasets, including data from NASA's Landsat, MODIS, Sentinel, and other global environmental datasets. GEE enables users to analyze trends, detect changes, and build predictive models using JavaScript or Python APIs.

How GEE is Useful for Our Project:

1. **Access to Satellite Imagery** – We can use GEE to obtain high-resolution satellite images for analyzing landfills, waste collection zones, and urban infrastructure.
2. **Geospatial Data Processing** – GEE's cloud-based environment allows efficient raster and vector data analysis without requiring extensive local computational resources.
3. **Environmental Analysis** – We can monitor waste collection efficiency, urban sprawl, and environmental impacts in real time.
4. **Integration with Python & Streamlit** – By using the Python API, we can integrate GEE with our Streamlit dashboard to visualize geospatial insights dynamically.

3.4 STREAMLIT

Streamlit is an open-source Python framework that simplifies the process of building interactive web applications for data visualization and machine learning models. Unlike traditional web frameworks, Streamlit allows users to turn Python scripts into web apps with minimal effort, eliminating the need for HTML, CSS, or JavaScript.

Key Features of Streamlit:

- **Real-time interactive widgets** – Users can interact with filters, sliders, and maps to explore data dynamically.
- **Automatic UI generation** – Streamlit automatically renders widgets and graphs based on the Python script, reducing development time.
- **Seamless integration with data libraries** – Works with Pandas, Matplotlib, Plotly, Folium, and Geopandas for advanced data visualization.
- **Live updates and caching** – Ensures fast rendering and smooth user experience.

How We Use Streamlit in Our Project:

1. **Visualizing Collection Routes** – We can use Streamlit and Folium to display interactive maps of landfill sites, collection zones, and truck routes.
2. **Analyzing Missed Pickups** – Streamlit can generate graphs and tables showing missed collections, allowing supervisors to identify problem areas.
3. **Comparing Waste Collection Metrics** – We can integrate data from GPS tracking, landfill tonnage (RAD), and recycling routes into a single dashboard for analysis.
4. **Dynamic Filtering** – Users can filter by district, waste type, or route ID to explore trends and optimize operations.

CONCLUSION

Each of these technologies plays a vital role in our project:

- **IDE** streamlines development and debugging.
- **Python environments** ensure dependency management and reproducibility.
- **Google Earth Engine** provides access to powerful geospatial analysis tools.
- **Streamlit** transforms complex data into an interactive and user-friendly web application.

By integrating these tools effectively, we create a **data-driven waste management system** that enhances operational efficiency, optimizes collection routes, and improves environmental sustainability.

4. REQUIREMENTS

4.1 SETTING UP ENVIRONMENT

1. Open the Telematics Folder in IDE
2. Open Terminal: On the taskbar, click on the terminal dropdown and click on New Terminal.
3. In the terminal, type the following to create a conda mamba environment:
   3.1. conda create -n streamlit_env
4. Activate the newly created environment:
   4.1. conda activate streamlit_env
5. In IDE, go to View > Command Palette > Select Interpreter, and choose the interpreter with 'streamlit_env' in the name.
6. Run the following command in the terminal to install dependencies:
   6.1. pip install streamlit streamlit_folium geopandas duckdb ee geemap
7. Create a Google Cloud Account if you don't already have one.
8. Create a New Project in the Google Cloud Console and give it a specific project name.
9. In VS Code, go to line 299 of your new_app.py file and update the project name:
   9.1. project = '<your project name>'
10. Enable the Google Earth Engine API in the Google Cloud Console for your project.
11. Register the Project with Google Earth Engine to link it to your project.
12. Run the Streamlit app by executing:
    12.1. streamlit run new_app.py
    12.2. If it doesn't run, try another port: streamlit run new_app.py --server.port 8080

4.2 DATA

Ensure your have the following files and folders:

- Files:
  - gps.db
    - **DuckDB database** containing several tables:
      - ARTS - data manually logged by supervisors such as driver assignment for truck/route
      - RAD - data collected from landfill scale such as tonnage measurements
      - GPS – records the real-time location and movement of collection trucks, including vehicle details, speed, timestamps, and spatial coordinates
  - routes.pqt

        - provides information on collection routes, including operational details, infrastructure metrics, and spatial attributes
    - new_app.py
        - Python script used to run the Streamlit application for data visualization and analysis.
    - Logo.png
        - Image file of Environmental Services department logo
    - pic1.png, pic2.png, pic3.png
        - Image files featuring trucks and San Diego logo

Each shapefile contains attributes such as **district, route ID, household count, load capacity, and spatial geometry** representing collection areas.

- **Landfill** – Polygon data representing landfill locations.
- **Org** – Organics collection routes.
- **org_htc** – Hard-to-collect organics collection routes.
- **Rec** – Recycling collection routes.
- **rec_htc** – Hard-to-collect recycling collection routes.
- **Ref** – Refuse (trash) collection routes.
- **ref_htc** – Hard-to-collect refuse collection routes.

For detailed dataset schemas, refer to **Appendix 5.2**.

4. NEXT STEPS

4.1 IMPROVEMENTS

Several enhancements can be made to improve both the functionality and user experience of the application:

- **Visual Design**: Refine the visual layout to make the application more cohesive, modern, and user-friendly by using consistent color schemes, spacing, and typography.
- **Expanded Data Analysis**: Incorporate more advanced data analytics, such as historical trends, missed pickup patterns, and predictive insights, rather than focusing solely on basic routing information.
- **Increased Interactivity**: Add features like filterable metrics, dynamic tooltips, and interactive charts to allow users to explore the data in more detail.
- **Enhanced Mapping**: Improve the map visualization of truck routes with clearer styling, better labeling, and the ability to view specific days, neighborhoods, or truck IDs.

- **Real-Time Data Integration**: Incorporate live data feeds (e.g., from GPS tracking or RAD systems) to reflect real-time truck locations, container fill levels, or active service requests.

These improvements would significantly enhance the value and usability of the application for decision-makers and field supervisors.

4.2 FUTURE WORK

Currently, the application is hosted on **Streamlit**, which is ideal for development and internal testing. However, to support wider access and real-world deployment, we can explore more robust hosting options:

- **Cloud Deployment**: Hosting the application on platforms such as AWS (Amazon Web Services), Google Cloud Platform, or Microsoft Azure would allow us to scale the app for larger user bases and ensure high availability. These platforms also offer built-in security, load balancing, and database integration.
- **Containerization with Docker**: Packaging the application in a Docker container would make it portable and easier to deploy across different environments, whether locally or in the cloud. This also improves consistency across development, testing, and production stages.
- **Integration with City Infrastructure**: For long-term use, the application could be embedded into existing internal portals or dashboards used by city departments, allowing seamless access for supervisors and analysts.
- **API Development**: Creating APIs to connect with real-time data sources (e.g., GPS databases, RAD systems, service request platforms) would enable continuous data updates and improve interactivity.
- **User Management and Authentication**: To support multiple users with different access levels (e.g., field staff vs. administrators), we can implement user authentication and role-based access control.

These steps would help move the application from a prototype to a fully operational tool that can be used across departments and adapted for broader public service applications.

5. LITERATURE REVIEW

5.1 ANALYSIS AND DESIGN DOCUMENT – ROUTE OPTIMIZATION SOLUTION INTEGRATION (CGI)

The Analysis and Design Document – Route Optimization Soultion Integration is a formally structured technical deliverable by CGI that outlines the integration of Routeware's route optimization system with the Environmental Services Department's (ESD) GIS environment. The document is organized into major sections and appendices, each serving a specific function within the software development and deployment lifecycle. Below is a detailed review focused on its section headings, content scope, and formatting approach:

1. PRELIMINARY ANALYSIS

This foundational section provides the **context**, **scope**, and **constraints** for the project.

- **1.1 Overview** – Introduces the operational need for integrating Routeware with ESD's GIS, driven by inefficiencies in current manual processes and regulatory expansion (e.g., Organics collection). It describes system architecture and integration goals, using clear and actionable language.
- **1.2 Assumptions** – Lists project constraints such as API-based data transfers, consistent primary keys, and staffing availability for UAT and deployment validation. This section adheres to standard systems design practices by identifying preconditions for success.
- **1.3 Constraints** – Outlines known limitations like environment availability and vendor staff timing, linking these to potential project delays.
- **1.4 Dependencies** (newly added) – Identifies critical system infrastructure (e.g., GISAPPSERVERDEV, ATLASQA) and software versioning (ArcGIS 10.8.1). This addition makes the project's technical foundation clearer and more auditable.

2. BUSINESS PROCESS

This section defines **business-level requirements** from a functional and organizational perspective.

- **2.1 High-Level Business Requirements** – Presented in a tabular format with columns for Req#, Requirement, Comments, and Priority, allowing traceability and prioritization.

4. QUALITY ASSURANCE

A brief but essential section covering **testing responsibilities** and phases, divided between CGI and client roles. It outlines acceptance standards, indicating the collaborative QA process used for deployment validation.

5. EFFORT ESTIMATE

This section presents **resource and cost modeling**, structured into three subsections:

- **5.1 Total GIS Integration Effort by Feature** – Lists tasks (e.g., Dev Consultation, Hypercare Support) with hour estimates and priority comments. This enables effective budgeting and workload distribution.
- **5.2 Total Project Management Effort** – Identifies 635 hours allocated to full-lifecycle PM, including Hypercare, and notes post-Hypercare disengagement dates. This precision supports contractual planning.
- **5.3 Total Cost by Role** – Although not fully visible in the provided content, this section likely completes the financial model with per-role cost allocations, a standard inclusion in technical project plans.

6. PROJECT TIMELINE

A planning section that visually defines the sequence of deliverables across phases, including UAT and deployment for each. This supports risk mitigation and alignment with organizational rollouts.

7. APPENDICES

The appendices provide critical technical artifacts and schema details to support implementation.

8. FORMATTING

The document is consistently formatted with:

- Numbered section headings for navigation and version control.
- Clear use of tables for requirements, assumptions, and effort estimates.
- Use of diagrams and appendices to visually support text-heavy sections.
- Progressive updates tracked in the Revision History, indicating structured document lifecycle management.

5.2 THE DESIGN AND IMPLEMENTATION OF A GARBAGE TRUCK FLEET MANAGEMENT SYSTEM

The design and implementation of a garbage truck fleet management system is a research article focused on the design and implementation of a Garbage Truck Fleet Management System (GTFMS) that integrates arrival time forecasting and an optimized power-saving mechanism for mobile applications. The study highlights how advances in cloud computing, mobile computing, and intelligent transportation systems (ITS) can be applied to improve urban waste collection services. This review outlines the structure of the original article, describes the main topics covered in each section, and includes notes on formatting such as the use of figures, graphs, and mathematical models.

1. ABSTRACT

- Summarizes the scope, methodology, and key findings of the study.
- Introduces the GTFMS architecture and highlights power efficiency and forecasting accuracy.

2. INTRODUCTION

- Discusses the context of ITS and mobile apps in public services.
- Establishes the motivation for developing a garbage truck app with real-time and predictive features.
- Briefly mentions forecasting methods and concerns about power usage.

3. LITERATURE REVIEW

Divided into three subsections:

- 3.1 Message Updating Mechanism
  - Reviews studies on optimizing update frequency and reducing power consumption.
- 3.2 Arrival Time Forecasting Method
  - Summarizes machine learning techniques such as regression, k-NN, SVM, and neural networks used for travel time prediction.
- 3.3 Cloud Computing
  - Explains the role of distributed computing in managing large datasets for real – time services.

3. DESIGN OF MANAGEMENT SYSTEM

- Describes the system’s three components: On-Board Units (OBUs), Fleet Management System(FMS), and Geographic Tracking Application (GTA).

5. MODEL OPTIMIZATION AND USER FEEDBACK

- Presents a six-step process based on user location and garbage truck proximity.
- 5.1 Arrival Time Forecasting Method
    - Details the use of weighted multiple linear regression models and training procedures.
- 5.2 Message Updating Mechanism
    - Describes how and when updates are triggered based on user settings and historical averages.
- 5.3 Case Study and Availability Rate Analysis
    - Includes a case study from Hsinchu City and performance metrics.
- 5.4 Cloud Computing for Forecasting
    - Explains how the system uses Hadoop and MapReduce to scale the forecasting process.

6. IMPLEMENTATION AND EVALUATION

- **6.1 Implementation:** Describes the deployment of the app on an Android device and how it interacts with user preferences.
- **6.2 Evaluation**: Analyzes forecasting accuracy using historical route data collected over sevenmonths.

6. CONCLUSION AND FUTURE WORK

- Summarizes the effectiveness of the proposed system in improving prediction accuracy and energy efficiency.
- Suggests areas for future enhancement, such as expanded datasets or algorithm refinements.

8. FORMATTING

- Use of Figures and Graphs

- Mathematical Models
- Consistent font, no color in the text
- Numbered headers.
- Reference Section

5.3 APPLICATION FOR IMAGE RECOGNITION IN WASTE MANAGEMENT

RecycleMate is a mobile application designed to help users identify recyclable items and locate nearby recycling facilities using image recognition and geolocation.

1. ABSTRACT

- Summarizes the scope and key findings of image recognition in recycling systems.
- Highlights the use of machine learning, particularly convolutional neural networks (CNNs), to classify recyclable materials.
- Emphasizes real-time classification accuracy and mobile integration.

2. INTRODUCTION

- Discusses the global challenge of improper waste sorting and declining recycling rates.
- Establishes the motivation for developing AI-powered tools to help individuals and municipalities sort waste accurately.
- Introduces image recognition as a scalable solution for public awareness and operational efficiency.

3. LITERATURE REVIEW

Divided into three subsections:

- **3.1 Image Classification Models**
    - Reviews common machine learning models used for image classification: CNNs, MobileNet, ResNet, and custom-trained AutoML Vision models.

  - Discusses training datasets such as TrashNet, TACO (Trash Annotations in Context), and proprietary datasets.
  - Emphasizes tradeoffs between model complexity and mobile deployment.
- **3.2 Mobile and Web Applications**
  - Summarizes several mobile applications, such as RecycleMate and Lasso, that allow users to scan waste items.
  - Highlights UI/UX considerations for real-time interaction.
  - Notes challenges in lighting conditions, image clarity, and model interpretability.
- **3.3 Integration with Recycling Infrastructure**
  - Discusses how systems can be integrated into Material Recovery Facilities (MRFs) or smart bins.
  - Outlines potential for automated sorting arms or robotic mechanisms.
  - Mentions pilot studies in urban cities exploring image-assisted bin classification.

4. DESIGN OF IMAGE RECOGNITION SYSTEM

- Describes a modular design comprising:
- Mobile application (user-facing front-end).
- Cloud-based classification service (e.g., Google AutoML, AWS Rekognition).
- Optional feedback module for model retraining.
- Notes asynchronous processing to minimize app delay.

5. MODEL OPTIMIZATION AND USER FEEDBACK

- **5.1 Data Augmentation and Preprocessing**
  - Details common techniques: rotation, background removal, contrast enhancement.
  - Notes importance of representative data for training and evaluation.
- **5.2 Active Learning and Retraining**
  - Describes how user corrections can be used to retrain models periodically.
  - Encourages a feedback loop between users and developers.
- **5.3 Performance Metrics**
  - Reports classification accuracy between 85–95% depending on the dataset and model.
  - Includes precision-recall analysis and confusion matrix for common material types (plastic, glass, paper, etc.).
- **5.4 Scalability in the Cloud**

  - Explains how services like Google Cloud Vision or AWS Lambda scale to handle multiple concurrent requests.
  - Discusses latency reduction strategies and cost tradeoffs.

6. IMPLEMENTATION AND EVALUATION

- **6.1 Implementation:**
  - Describes deployment on Android and iOS using cross-platform tools like React Native or Flutter.
- Notes integration with image APIs, real-time feedback, and camera handling.
- **6.2 Evaluation:**
  - Evaluates the model with a test set and real-world user trials.
  - Discusses user engagement data, error rates, and usability surveys.

7. CONCLUSIONS AND FUTURE WORK

- Concludes that image recognition significantly enhances user participation and accuracy in recycling.
- Recommends future work in:
  - Expanding to more item types.
  - Improving edge-device processing.
  - Integrating with location services to offer disposal sites.
  - Exploring multilingual or voice-based interfaces.

7. RELATIONSHIP TO OUR WORK

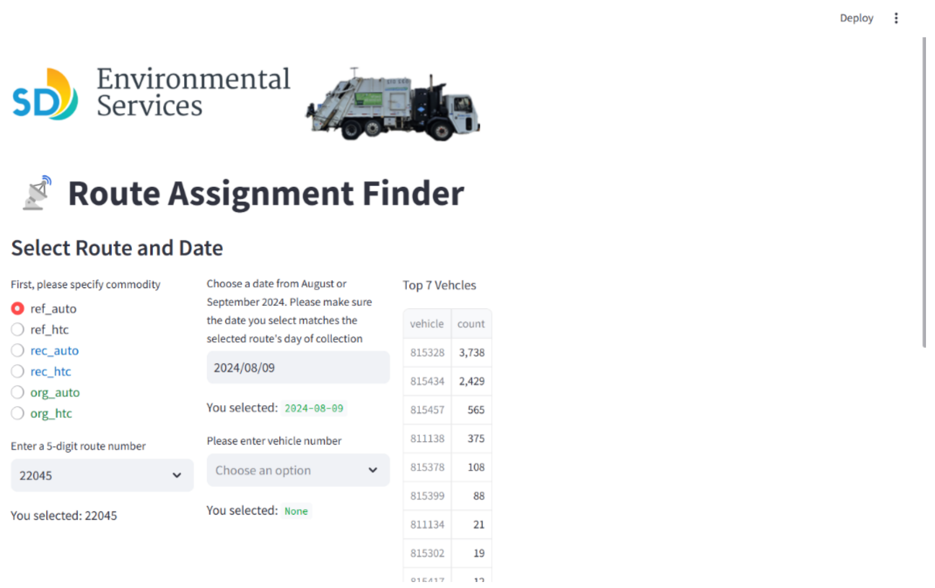
- While RecycleMate focuses on individual user education and behavior change, our project targets municipal-level operational efficiency.
- Shared goals:
  - Both aim to improve waste management systems through technology-driven solutions.
  - Both emphasize location-based services, with RecycleMate using geolocation to find facilities, and Route Assignment Finder leveraging geospatial data for route analysis.
- Key differences:

- RecycleMate uses computer vision (image recognition) to classify waste items, while Route Assignment Finder uses spatial joins and routing algorithms for logistical planning.
- RecycleMate is citizen-facing, promoting recycling awareness, while Route Assignment Finder is staff-facing, enhancing service delivery operations.

- Potential integration:
  - Data from RecycleMate (e.g., recycling drop-off demand hotspots) could inform route planning in Route Assignment Finder.
  - Our project could expand to include interactive tools for public education, similar to RecycleMate, using dashboard elements.

6. APPENDIX

6.1 USER INTERFACE



Route Assignment Finder

Select Route and Date

First, please specify commodity

[✓ ref_auto]
ref_htc
rec_auto
rec_htc
org_auto
org_htc

Enter a 5-digit route number

22045

You selected: 22045

Choose a date from August or September 2024. Please make sure the date you select matches the selected route's day of collection

2024/08/09

You selected: `2024-08-09`

Please enter vehicle number

Choose an option

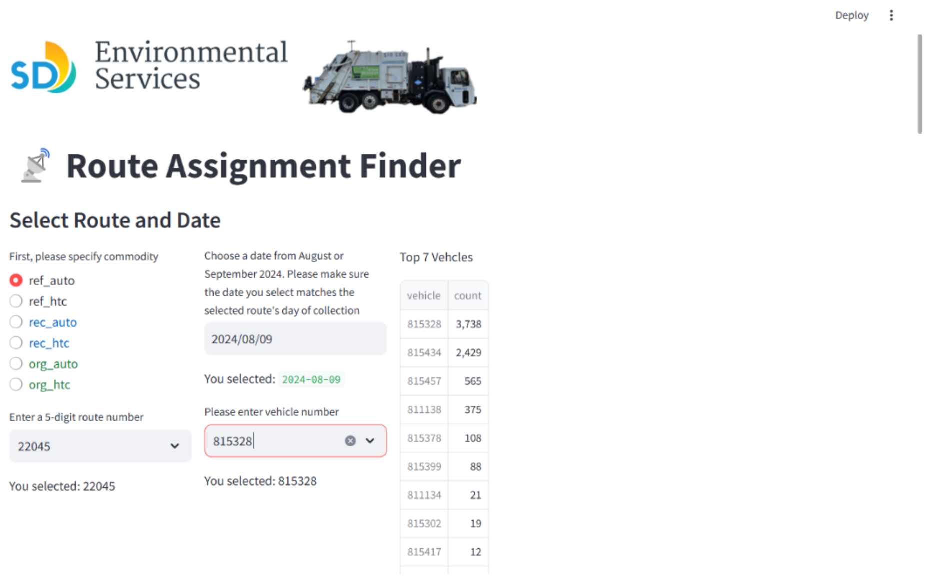
You selected: `None`

Top 7 Vehcles

vehicle	count
815328	3,738
815434	2,429
815457	565
811138	375
815378	108
815399	88
811134	21
815302	19

Figure 1: Welcome Page

This screen allows the user to select a route and date to begin the assignment process.



Route Assignment Finder

Select Route and Date

First, please specify commodity

- [✓ ref_auto]
- ref_htc
- rec_auto
- rec_htc
- org_auto
- org_htc

Enter a 5-digit route number

22045

You selected: 22045

Choose a date from August or September 2024. Please make sure the date you select matches the selected route's day of collection

2024/08/09

You selected: `2024-08-09`

Please enter vehicle number

815328

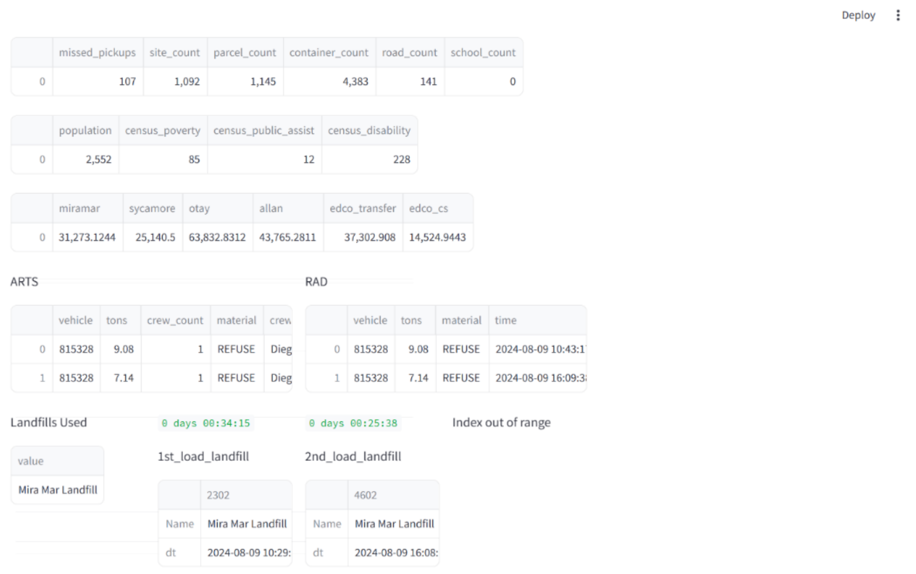
You selected: 815328

Top 7 Vehcles

vehicle	count
815328	3,738
815434	2,429
815457	565
811138	375
815378	108
815399	88
811134	21
815302	19
815417	12

Figure 2: Select Vehicle Number

Route Selection Example: Once a specific route and date are selected, the app displays the top 7 trucks assigned to that route for that day.



	missed_pickups	site_count	parcel_count	container_count	road_count	school_count
0	107	1,092	1,145	4,383	141	0

	population	census_poverty	census_public_assist	census_disability
0	2,552	85	12	228

	miramar	sycamore	otay	allan	edco_transfer	edco_cs
0	31,273.1244	25,140.5	63,832.8312	43,765.2811	37,302.908	14,524.9443

ARTS

	vehicle	tons	crew_count	material	crew
0	815328	9.08	1	REFUSE	Dieg
1	815328	7.14	1	REFUSE	Dieg

RAD

	vehicle	tons	material	time
0	815328	9.08	REFUSE	2024-08-09 10:43:1
1	815328	7.14	REFUSE	2024-08-09 16:09:3

Landfills Used

value
Mira Mar Landfill

`0 days 00:34:15`

1st_load_landfill

	2302
Name	Mira Mar Landfill
dt	2024-08-09 10:29:

`0 days 00:25:38`

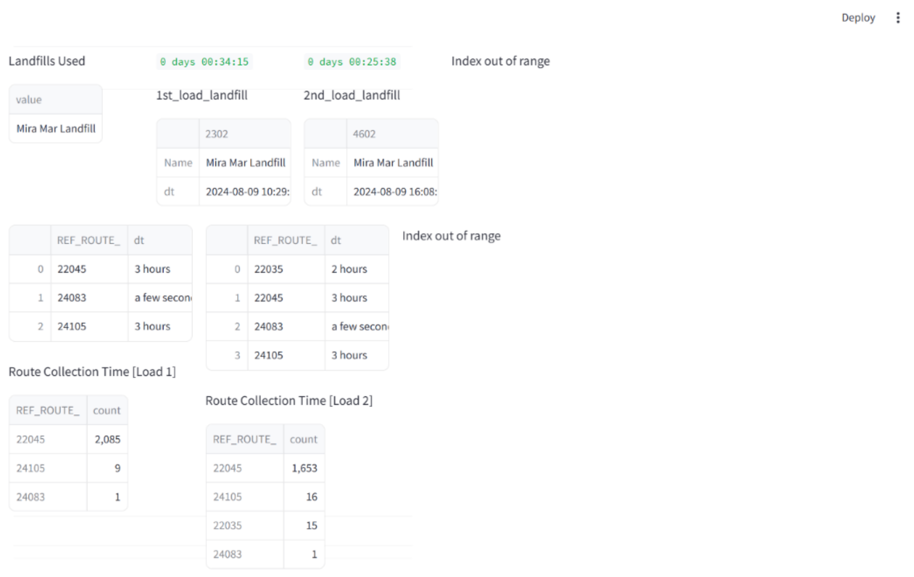
2nd_load_landfill

	4602
Name	Mira Mar Landfill
dt	2024-08-09 16:08:

Index out of range

Figure 3: Route DataFrame

After selecting a vehicle and date, the app presents route statistics, including missed pickups, site count, population served, and landfill data (ARTS/RAD).



Landfills Used

value
Mira Mar Landfill

`0 days 00:34:15` `0 days 00:25:38`

1st_load_landfill

	2302
Name	Mira Mar Landfill
dt	2024-08-09 10:29:

2nd_load_landfill

	4602
Name	Mira Mar Landfill
dt	2024-08-09 16:08:

Index out of range

	REF_ROUTE_	dt
0	22045	3 hours
1	24083	a few secon
2	24105	3 hours

	REF_ROUTE_	dt
0	22035	2 hours
1	22045	3 hours
2	24083	a few secon
3	24105	3 hours

Index out of range

Route Collection Time [Load 1]

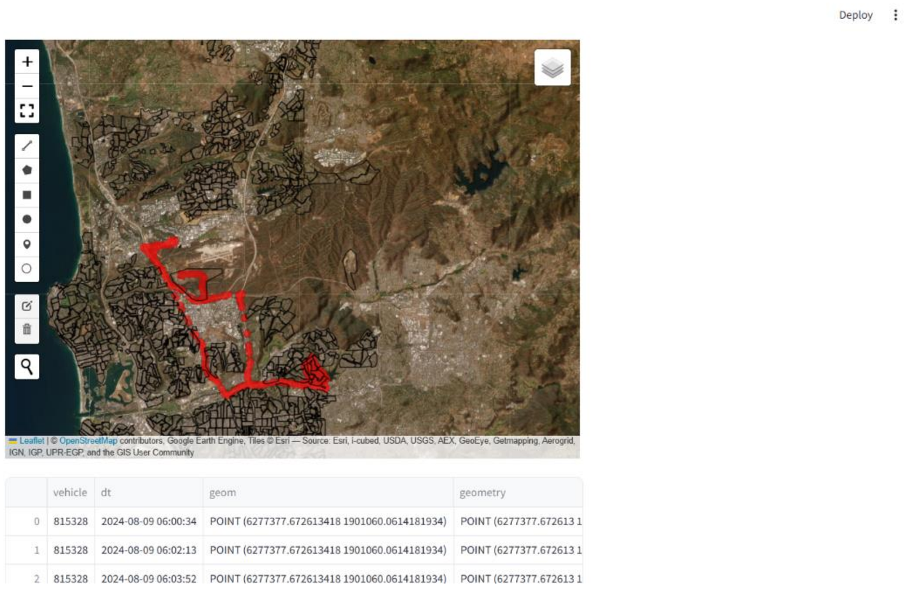
REF_ROUTE_	count
22045	2,085
24105	9
24083	1

Route Collection Time [Load 2]

REF_ROUTE_	count
22045	1,653
24105	16
22035	15
24083	1

Figure 4: Landfill DataFrame

This table displays the landfills used for the selected route along with time spent en route and at each landfill.

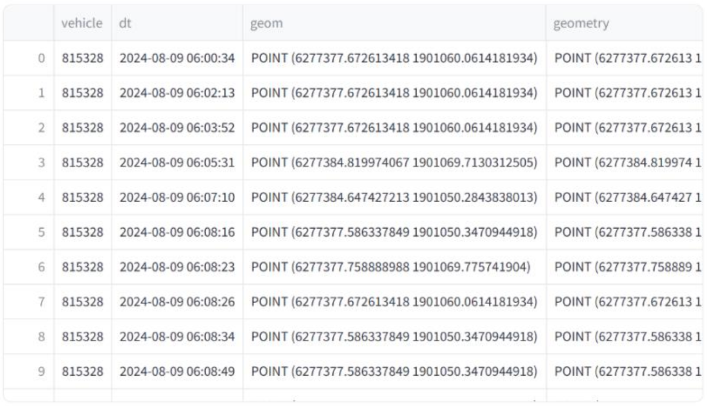


Leaflet | © OpenStreetMap contributors, Google Earth Engine, Tiles © Esri — Source: Esri, i-cubed, USDA, USGS, AEX, GeoEye, Getmapping, Aerogrid, IGN, IGP, UPR-EGP, and the GIS User Community

	vehicle	dt	geom	geometry
0	815328	2024-08-09 06:00:34	POINT (6277377.672613418 1901060.0614181934)	POINT (6277377.672613 1
1	815328	2024-08-09 06:02:13	POINT (6277377.672613418 1901060.0614181934)	POINT (6277377.672613 1
2	815328	2024-08-09 06:03:52	POINT (6277377.672613418 1901060.0614181934)	POINT (6277377.672613 1

Figure 5: Route Map

The selected route is highlighted on an interactive map, providing a clear visual of the route path.



	vehicle	dt	geom	geometry
0	815328	2024-08-09 06:00:34	POINT (6277377.672613418 1901060.0614181934)	POINT (6277377.672613 1
1	815328	2024-08-09 06:02:13	POINT (6277377.672613418 1901060.0614181934)	POINT (6277377.672613 1
2	815328	2024-08-09 06:03:52	POINT (6277377.672613418 1901060.0614181934)	POINT (6277377.672613 1
3	815328	2024-08-09 06:05:31	POINT (6277384.819974067 1901069.7130312505)	POINT (6277384.819974 1
4	815328	2024-08-09 06:07:10	POINT (6277384.647427213 1901050.2843838013)	POINT (6277384.647427 1
5	815328	2024-08-09 06:08:16	POINT (6277377.586337849 1901050.3470944918)	POINT (6277377.586338 1
6	815328	2024-08-09 06:08:23	POINT (6277377.758888988 1901069.775741904)	POINT (6277377.758889 1
7	815328	2024-08-09 06:08:26	POINT (6277377.672613418 1901060.0614181934)	POINT (6277377.672613 1
8	815328	2024-08-09 06:08:34	POINT (6277377.586337849 1901050.3470944918)	POINT (6277377.586338 1
9	815328	2024-08-09 06:08:49	POINT (6277377.586337849 1901050.3470944918)	POINT (6277377.586338 1

Figure 6: GPS points DataFrame

This view tracks trucks via GPS, showing their locations and the duration spent at each point.

6.1 DATASET SCHEMA

COLUMN NAME	DATA TYPE
ROUTE#	BIGINT
TRANSACTION #	BIGINT
COLL_DT	VARCHAR
RPT_DT	VARCHAR
LD_DT_TM	TIMESTAMP
CO_FLEET_ID	BIGINT
CREW_CNT	BIGINT
TONS	DOUBLE
BLUE_ORANGE_WEEK_CD	VARCHAR
DAY_OF_WK	BIGINT
MATL_TYP_DESC	VARCHAR
CREW NAME	VARCHAR
NBR_LOADS	BIGINT
DUMP LOCATION	VARCHAR

Figure 7: arts Dataset Schema

COLUMN NAME	DATA TYPE
TRAN_NUM	BIGINT
LD_DT_TM	TIMESTAMP_NS
RPT_DT	VARCHAR
TONS	DOUBLE
CO_FLEET_ID	VARCHAR
MATL_TYP_SD	BIGINT
MATL_TYP_DESC	VARCHAR

LANDFILL_DEST_DESC	BIGINT

Figure 8: rad Dataset Schema

COLUMN NAME	DATA TYPE
EquipmentNum	VARCHAR
VIN	VARCHAR
LicenseEPlateNum	VARCHAR
Speed	BIGINT
Latitude	DOUBLE
Longitude	DOUBLE
DateTimeUTC	VARCHAR
DateTimeLocal	VARCHAR
geom	GEOMETRY

Figure 9: gps_xy Dataset Schema

COLUMN NAME	DATA TYPE
route	object
orange_blue	object
supervisor	object
district	object
site_count	int64
missed_pickups	int64
parcel_count	int64
parcel_st	object
parcel_mf_4	object
parcel_mf_5	object
parcel_other	object
container_count	int64

container_ref	object
container_rec	object
container_org	object
container_2014	object
road_count	int64
road_1	object
road_2	object
road_alley	object
school_count	int64
school_middle	object
school_high	object
miramar	float64
…	…
index	object
Shape_length	float64
Shape_Area	float64

Figure 10: routes.pqt Schema

COLUMN NAME	DATA TYPE
Shape_Leng	float64
Shape_Area	float64
Name	object
Commodity	int64
Organic	int64
Address	object
geometry	geometry

Figure 10: landfill Dataset Schema

COLUMN NAME	DATA TYPE
REF_DISTRI	object
REF_SUP_ID	object
REF_DAY_OF	int64
REF_ROUTE_	object
REFROUTE	object
REF_RTE_SE	object
REF_LOAD	int64
REF_CTR_NM	object
REF_MAP_IN	int64
REF_RT_IDX	object
REF_SC_IDX	object
ShapeSTAr	float64
Shape_STLe	float64
geometry	geometry

Figure 11: ref_auto Dataset Schema

COLUMN NAME	DATA TYPE
REF_DISTRI	object
REF_SUPER_	object
REF_DAY_OF	int64
REF_ROUTE_	object
REFROUTE	object
REF_RTE_SE	object
REF_LOAD	int64

REC_CTR_NM	object
REF_MAP_IN	int64
ShapeSTAr	float64
Shape_STLe	float64
geometry	geometry

Figure 12: ref_htc Dataset Schema

COLUMN NAME	DATA TYPE
REC_DISTRI	object
REC_SUP_ID	object
REC_WEEK_C	object
REC_DAY_OF	int64
REC_ROUTE_	object
RECROUTE	object
REC_CTR_NM	object
REC_LOAD	int64
REC_STAT	object
RECYCLE_DT	object
REC_MAP_IN	int64
ShapeSTAr	float64
Shape_STLe	float64
geometry	geometry

Figure 13: rec_auto Dataset Schema

COLUMN NAME	DATA TYPE
REC_DISTRI	object

REC_SUPER_ID	object
REC_WEEK_C	object
REC_DAY_OF	int64
REC_ROUTE_	object
RECROUTE	object
REC_CTR_NM	object
REC_RTE_SE	object
REC_LOAD	int64
REC_STAT	object
RECYCLE_DT	object
REC_MAP_IN	int64
ShapeSTAr	float64
Shape_STLe	float64
geometry	geometry

Figure 14: rec_htc Dataset Schema

COLUMN NAME	DATA TYPE
ORG_DISTRI	object
ORG_SUPER_	object
ORG_DAY_OF	int64
ORG_ROUTE_	object
ORGROUTE	object
Draft	object
Draft_Date	object
ORGROUTE	object
House_Coun	int64
ORG_AUTO_S	object
ORG_RTE_SE	object
ORG_LOAD	int64

ORG_CTR_NM	object
ORG_DT	object
ORG_MAP_IN	int64
R_IDX	object
ShapeSTAr	float64
Shape_STLe	float64
geometry	geometry

Figure 15: org_auto Dataset Schema

COLUMN NAME	DATA TYPE
ORG_DISTRI	object
ORG_SUPER_	object
ORG_DAY_OF	int64
ORG_ROUTE_	object
ORGROUTE	object
House_Coun	int64
ORG_CTR_NM	object
ORG_LOAD	int64
ShapeSTAr	float64
Shape_STLe	float64
geometry	geometry

Figure 16: org_htc Dataset Schema

7. REFERENCES

https://www.researchgate.net/publication/303788309_The_design_and_implementation_of_a_garbage_truck_fleet_management_system

https://cityofsandiego-my.sharepoint.com/query?q=cgi&searchScope=all&id=%2Fpersonal%2Fmalsheikhyas%5Fsandiego%5Fgov%2FDocuments%2FDRIVE%2FH%2F072724%2FAdmin%2FRouting%2FT1DMND0106445%2FCOSD%5FADD%5F42174%5FT1DMND0106445%5FRoutewareIntegration%5F05112023%2Epdf&listurl=%2Fpersonal%2Fmalsheikhyas%5Fsandiego%5Fgov%2FDocuments&parentview=7