

Smart Trash Can Management System



MIS 6308.001

Professor Srinivasan Raghunathan

Group #1

Anmol Vyas, Aparna Satheesh,
Ganesa Parmar, Shalini Singh,
Sumeet Singh

Table of Contents

Project Presentation YouTube Link	3
Executive Summary	3
Problem Statement	4
Problem	4
Objectives	4
Scope	4
Future Enhancements	4
BPMN Diagram	6
Context Diagram	8
Use Case Diagram	9
Use Case Descriptions	10
Use Case Name: Find Trash Can Information	10
Use Case Name: Process Trash Capacity	11
Use Case Name: Send Trash Collection Notification	12
Use Case Name: Compare Trash Can Data	13
Use Case Name: Recommend Trash Can Placement	14
Use Case Name: Register Member	15
Use Case Name: Authorize Login	16
Use Case Name: Report Problem	18
Data Dictionary	19
Class Diagram	22
Complete Class Diagram	23
Sequence Diagram	24
Functional Specification	27
Interface Design	28
Database Design	33
Software Design	35
Project Management Deliverables	44
Meeting Timeline	44
Minutes of Meetings	45
References	51

1. Project Presentation YouTube Link

https://www.youtube.com/watch?v=jhJgJRRoB_E&feature=youtu.be

2. Executive Summary

In some places, it has been observed that the trash cans are overflowing with trash, and they stay full until their designated cleaning time. As a result, this problem leads to pollution and sometimes the spread of diseases.

Therefore, our group introduces the Smart Trash Can Management System, which is a cost effective and efficient automated IOT based system whose main objective is to resolve problems for trash collection and management by using ultrasonic sensors to monitor the fill capacity percentage of every trash can. Moreover, the system also provides recommendations for where there is a need of a new trash can so as to minimize the cost and human effort.

The Smart Trash Can Management System will allow users and members to view the amount of trash that is present in a trash can near their location by accessing the website. The user can then decide where to dispose their trash based on the amount of trash they have by looking at the system which will provide the information about the trash can fill capacity. As a result, a user will save a lot of time and effort.

The smart trash cans consist of an ultrasonic sensor that measures the level of trash in the trash can. These sensors are interfaced with a Node Microcontroller unit that transfers the trash fill capacity data to the trash can data file of the system. The Smart Trash Can Management System will automatically send a notification to the trash management if the fill capacity percentage of the trash can reaches the threshold of 80% or above. As a result, the trash cans can be prevented from getting overflowed.

The Smart Trash Can Management system also provides recommendations for new trash can placement by analyzing the trash collection frequency data of the current and past 6 months. The system calculates the current 6 months' cost for each trash can and that cost is compared to its cost of the last 6 months. Afterwards, the system selects the top 3 most costly trash cans from all of the trash cans that have a current individual cost that is 1.5 times more than their cost of the last 6 months. The locations of these top 3 trash cans are sent to trash management as a recommendation for a new trash can to be placed near their locations. By placing a new trash can, the system can reduce the cost incurred by trash management for sending their janitors an excessive number of times.

3. Problem Statement

Problems:

- 1) The current trash collecting system has no way to track if a trash can is filled or not, which causes the janitor to personally go around checking each trash can. This process is thus inefficient and time consuming.
- 2) The current system does not have a way to allow residents who need to throw trash away see which trash can is full or not. As a result, some trash cans are barely used while others are overflowing with trash.
- 3) The current system does not have a platform available to determine the best location for future trash can placement.

Objectives:

- 1) Trash Management will be notified if a trash can is full or empty through the platform so that they can work more efficiently
- 2) The residents can view the percentage of the trash in the trash can that'll help them to save time and energy
- 3) The platform will collect data about how frequently the trash can gets full so that the system can send out a recommendation to help with determining which location to place future trash cans. This process will save money as placing a new trash can will be less costly than having a janitor empty a trash can an excessive number of times

Scope:

- 1) The goal of this project would be to improve a company's trash collection, trash disposal, and trash can placement
- 2) This new system can be implemented in many other locations such as schools and malls
- 3) This project requires manpower to install sensors into the trash cans and develop a website
- 4) This system requires database servers to store each member's information and each trashcan's information

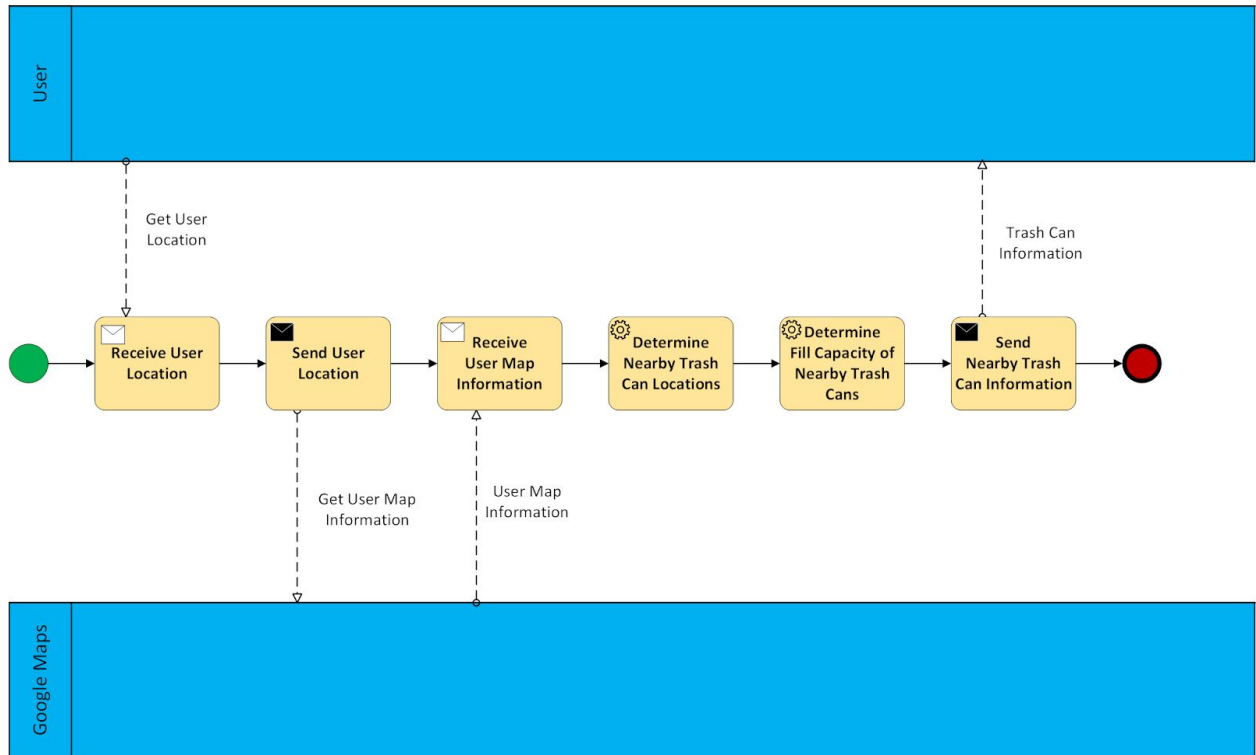
Future Enhancement:

- 1) We can add sensors and other hardware components to the recyclable trash cans so that we can check their fill capacity and compress the trash every time it nears a fill capacity of 80% to make more room in the trash can

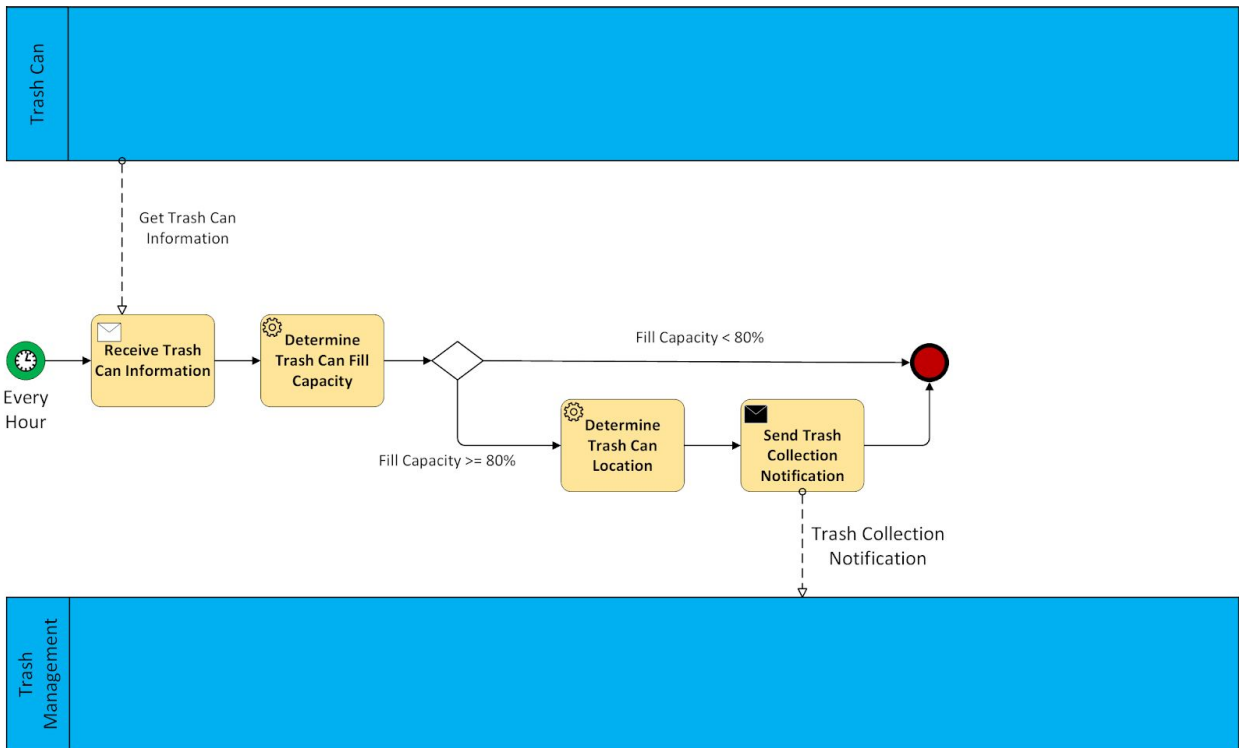
- 2) We can use solar powered batteries in the sensors, which will reduce the cost of powering the sensors as we have an eco-friendly and renewable source of energy.

4. BPMN Diagram

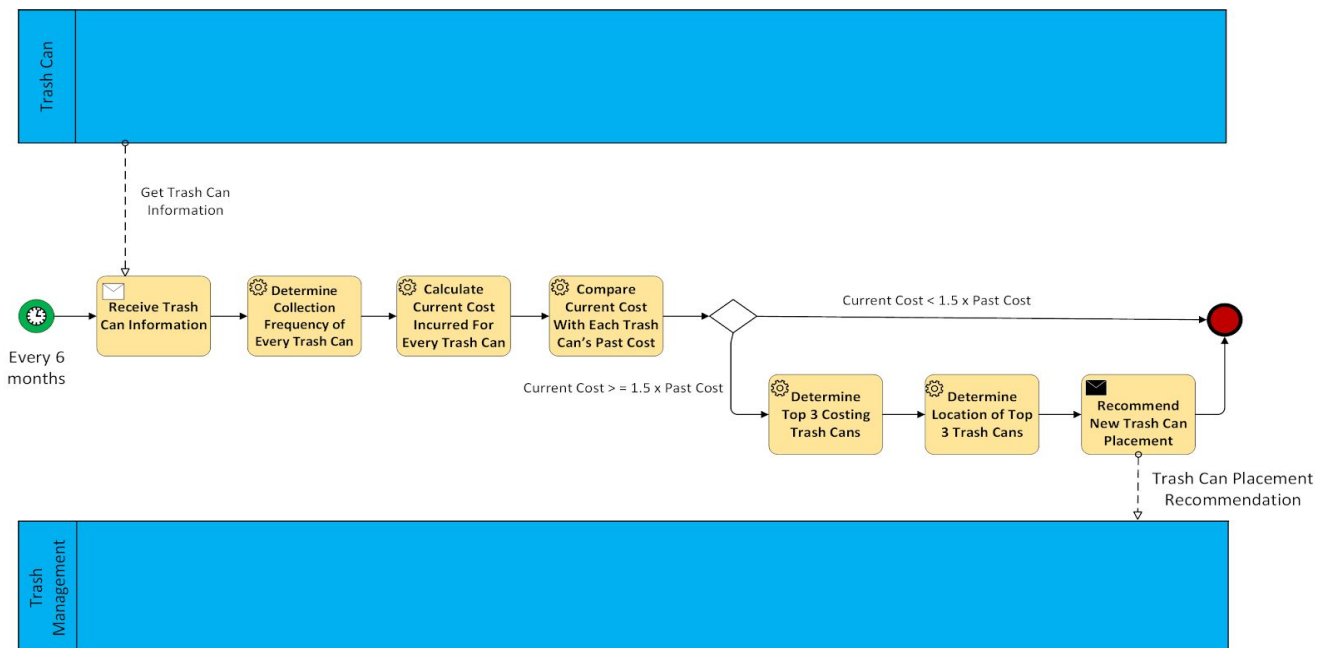
1. Find Trash Can Information



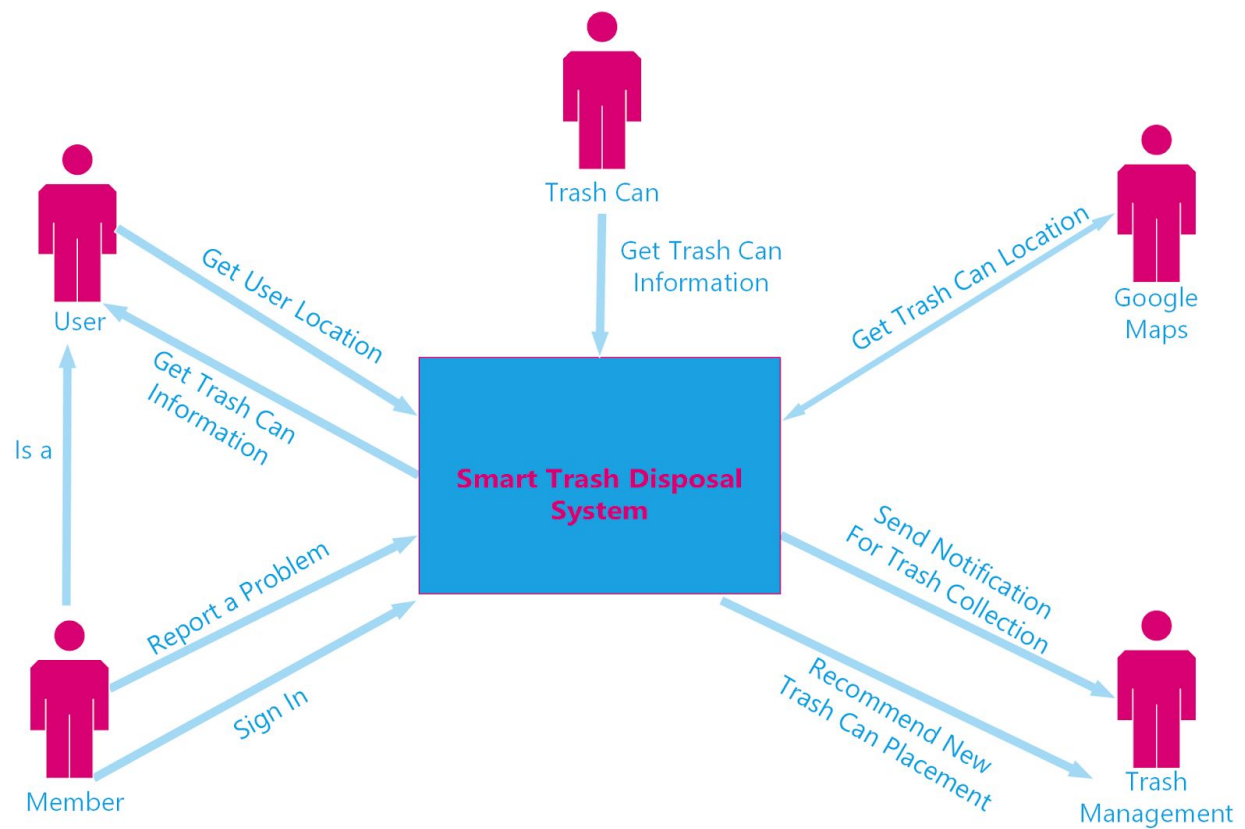
2. Send Trash Can Notification



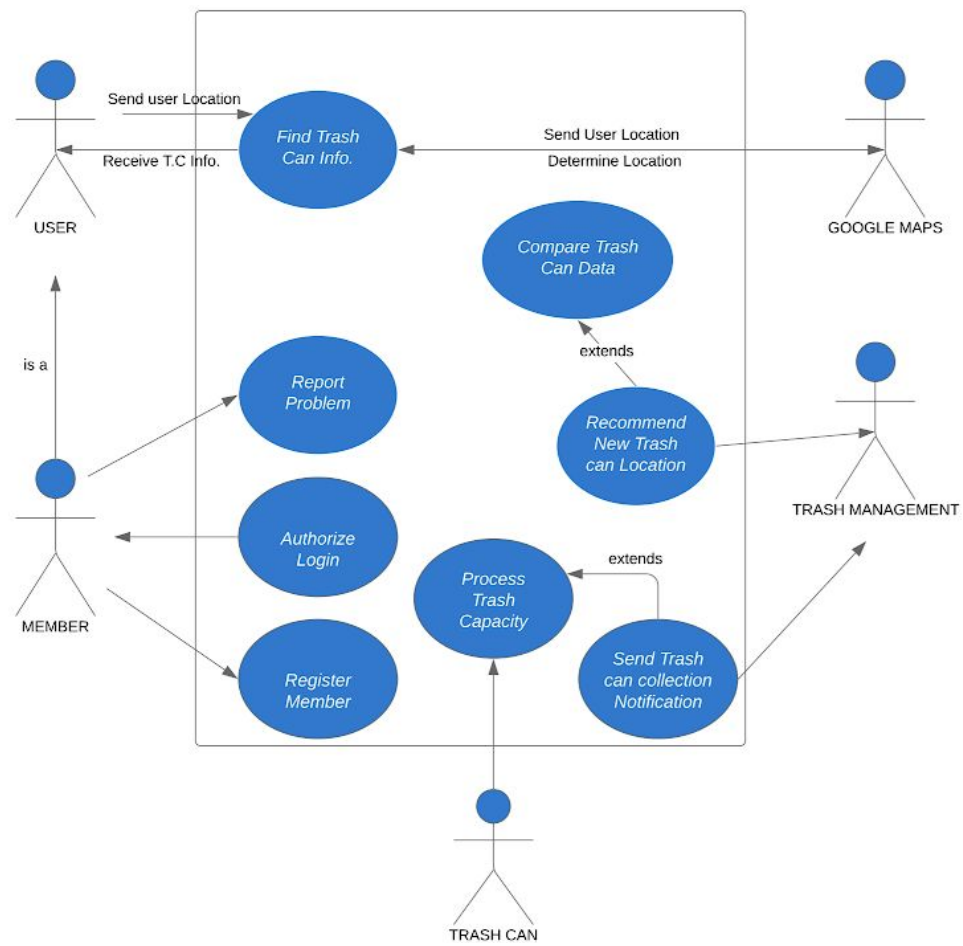
3. Recommend New Trash Can Placement



5. Context Diagram



6. Use Case Diagram



7. Use Case Descriptions

Use Case Description 1:

Use Case Name: Find Trash Can Information
Primary Actor: User/Member
Stakeholders: User/Member
Description: A User/Member wants to find out capacity of trash cans nearby so they can decide which trash can is best suited and has ample space that matches their trash disposal amount
Trigger: A User/Member clicks on Find My Can button
Relationships: Includes: Extends:
Normal flow of events: 1. <u>User location</u> is obtained 2. <u>Map Info</u> is retrieved using Google Maps 3. System determines the nearby <u>trash cans locations</u> using trash can data file 4. System determines the nearby trash cans' current <u>trash can fill capacity</u> , <u>trash can capacity available</u> , and <u>trash can color</u> using trash can data file 5. Displays the <u>nearby trash cans info</u> and each trash can's <u>trash can info</u>

Exceptional flow of events:

3A. If a trash can is not found near the given user location, display “No trash can found” message

Use Case Description 2:**Use Case Name:** Process Trash Capacity**Primary Actor:** System**Stakeholders:** Trash Can

Description: System receives the trash can fill capacity data from trash can and it updates the fill capacity of the trash can in the trash can data file. If the trash can fill capacity is 80% or more, then the system executes the Send Trash Collection Notification use case.

Trigger: Trash can sends its fill capacity data to the system at an interval of every hour

Relationships:**Includes:****Extends:**

Normal flow of events:

1. System receives the trash can fill capacity data from trash can every hour
2. System updates the trash can data file with the trash can fill capacity data received by trash can
3. System checks if the trash can fill capacity data of any trash can is 80% or greater
4. Use case Send Trash Collection Notification is executed

Exceptional flow of events:

3A. If fill capacity of the trash can is less than 80%, no action is taken by the system

Use Case Description 3:

Use Case Name: Send Trash Collection Notification
Primary Actor: System
Stakeholders: Trash Management
Description: A notification message is sent to trash management when a trash can's fill capacity reaches 80% or above
Trigger: A trash can reaches the capacity of 80%

Relationships: Includes: Extends: Process Trash Capacity
Normal flow of events: 1. System retrieves the <u>trash can location</u> from trash can data file of the trash cans that have a <u>trash can fill capacity</u> of more than 80% 2. System will send a <u>notification</u> to the trash management that displays the <u>notification ID</u> , <u>notification type</u> , <u>trash can ID</u> , and <u>trash can location</u>
Exceptional flow of events:

Use Case Description 4:

Use Case Name: Compare Trash Can Data
Primary Actor: System
Stakeholders: Trash Management
Description: System compares the cost incurred for individual trash can to its past 6 months cost data and determines which 3 trash cans contribute the highest to the cost incurred by trash management
Trigger: periodic event that takes place every 6 months

Relationships:**Includes:****Extends:****Normal flow of events:**

1. Every 6 months, the system calculates the individual current cost incurred by trash management for every trash can based on the trash can's collection frequency available in the collection frequency data file.
2. System compares the current six months cost incurred data against the previous 6 months cost incurred data from cost data file
3. System determines if the cost incurred for a trash can in the current 6 months is 50% more than the cost it incurred for the last 6 months
4. System determines the top 3 trash cans that cost the most and gets their locations using the trash can data file

Exceptional flow of events:

- 3A. If the cost incurred for any trash can in the current 6 months has not increased by more than 50% of the cost it incurred for the last 6 months, then the system does nothing

Use Case Description 5:**Use Case Name: Recommend New Trash Can Location****Primary Actor:** System**Stakeholders:** Trash Management

Description: Based on the result that system gets from Compare Trash Can Data use case, the system recommends the new trash can placement location to the trash management
Trigger: when system gets the locations of the top 3 trash cans
Relationships: Includes: Extends: Compare Trash Can Data
Normal flow of events: 1. System receives the <u>locations</u> of the top 3 trash cans from Compare Trash Can Data use case 2. System sends a recommendation to trash management to place a new trash can near the top 3 <u>trash can locations</u>
Exceptional flow of events:

Use Case Description 6:

Use Case Name: Register Member
Primary Actor: Member

Stakeholders: Member
Description: User can register themselves as member by filling the registration form on the website
Trigger: when user clicks on the registration tab on website
Relationships: Includes: Extends:
Normal flow of events: 1. User clicks on the registration tab on website 2. User enters <u>name</u> , <u>address</u> , and <u>DOB</u> into the member registration form to be stored into member data file 3. System asks the user to provide <u>username</u> and <u>password</u> which will be stored into member data file 4. System registers the user as a member after above procedure and gives the new member a <u>member ID</u>
Exceptional flow of events:

Use Case Description 7:

Use Case Name: Authorize login

Primary Actor: Member
Stakeholders: Member
Description: Only a registered member can login the website with their valid username and password.
Trigger: when the member enters the valid username and password
Relationships: Includes: Extends:
Normal flow of events: <ol style="list-style-type: none"> 1. System asks the member to enter their valid <u>username</u> and <u>password</u> in order to login to their user account. 2. System validates the <u>username</u> and <u>password</u> entered by the member using member data file 3. System allows the member to access their account if the <u>username</u> and <u>password</u> is validated
Exceptional flow of events: <p>2A. If a member does not give valid <u>username</u> and <u>password</u> when logging into their user account, a message for invalid <u>username</u> and <u>password</u> is sent to the user.</p>

Use Case Description 8:

Use Case Name: Report Problem
Primary Actor: Member
Stakeholders: Member
Description: Only a registered member can login the website and report a problem
Trigger: when the member logs in to the website and clicks on report problem tab.
Relationships: Includes: Authorize Login Extends:
Normal flow of events: 1. System asks the member to enter their valid <u>username</u> and <u>password</u> in order to login to their user account. 2. System allows the member to access their account with valid <u>username</u> and <u>password</u> after authorizing their login 3. Members clicks on report problem tab which pulls up a form displaying the <u>problem ID</u> , <u>member ID</u> , and <u>trash can ID</u> 4. Member gives the <u>problem description</u> and clicks on submit reports button.
Exceptional flow of events:

8. Data Dictionary

The data dictionary used is given below:

Use Case 1: Find Trash Can Information

User Location = [Zipcode | City + State | GPS location]

Map Info = Data Element

Trash Can Capacity Available = 100 - Trash Can Fill Capacity

Trash Can Color = [Green | Yellow | Red]

Nearby Trash Cans Info = 0{Trash Can ID + Trash Can Location + Trash Can Color + Trash Can Fill Capacity}

Trash Can Info = 0{Trash Can ID + Trash Can Fill Capacity + Trash Can Capacity Available}

Trash Can Data File = Trash Can ID + Trash Can Fill Capacity + Trash Can Capacity Available + Trash Can Color + Trash Can Location

Use Case 2: Process Trash Capacity

Trash Can Fill Capacity = Data Element

Use Case 3: Send Trash Collection Notification

Trash Can Location = Data Element

Trash Can Fill Capacity = Data Element

Notification ID = Data Element

Notification Type = Data Element

Trash Can ID = Data Element

Notification = Notification ID + Notification Type + Trash Can ID + Trash Can Location

Use Case 4: Compare Trash Can Data

Cost = Data Element

Collection Frequency = Data Element

Collection Frequency Data File = Trash Can ID + Notification ID + Collection Start Date
+ Collection End Date + Collection Frequency

Cost Data File = Trash Can ID + Cost + Start Date + End Date

Trash Can Location = Data Element

Use Case 5: Recommend New Trash Can Location

Trash Can Location = Data Element

Use Case 6: Register Member

Member Name = Data Element

Member Address = Data Element

Member DOB = Data Element

Username = Data Element

Password = Data Element

Member Data File = Member ID + Member Name + Member Address + Member DOB +
Username + Password

Use Case 7: Authorize login

Username = Data Element

Password = Data Element

Member ID = Data Element

Member Data File = Member ID + Member Name + Member Address + Member DOB +
Username + Password

Use Case 8: Report Problem

Username = Data Element

Password = Data Element

Problem ID = Data Element

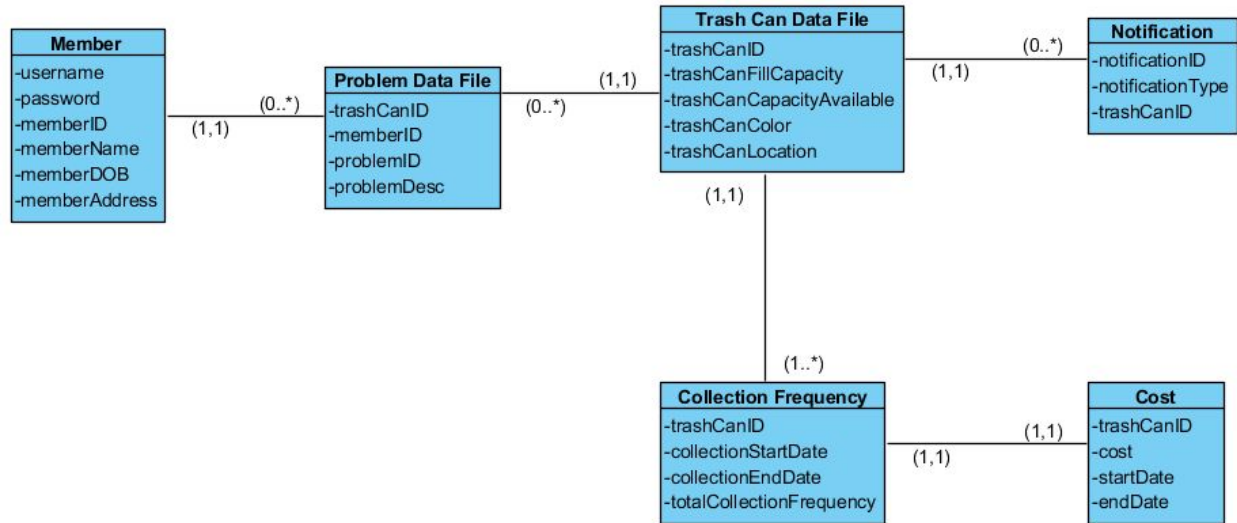
Trash Can ID = Data Element

Problem ID = Data Element

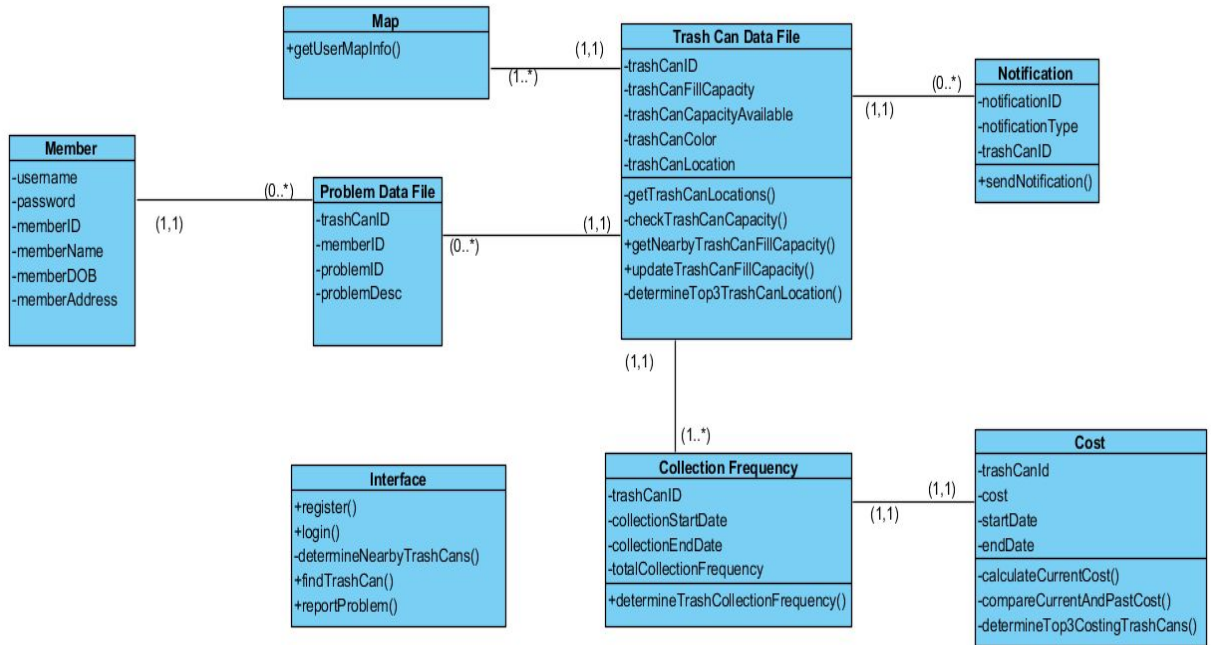
Problem Description = Data Element

Problem Data = Problem ID + Member ID + Trash Can ID + Problem Description

9. Class Diagram without Methods

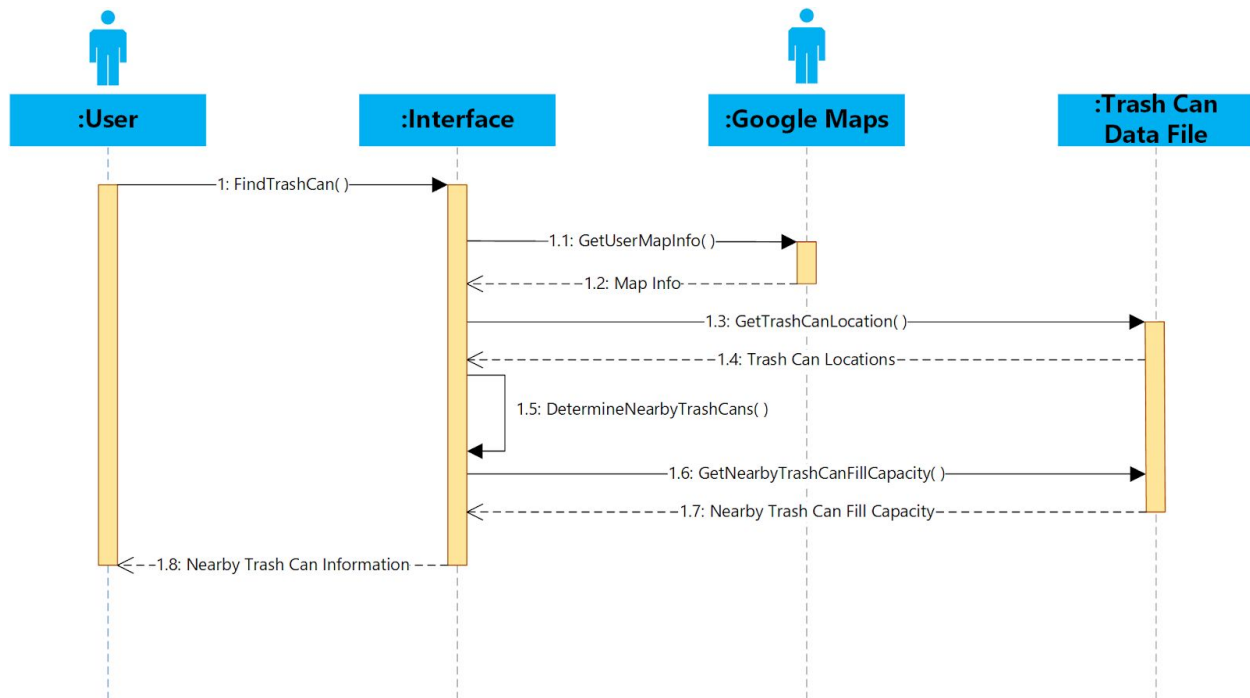


10. Complete Class Diagram

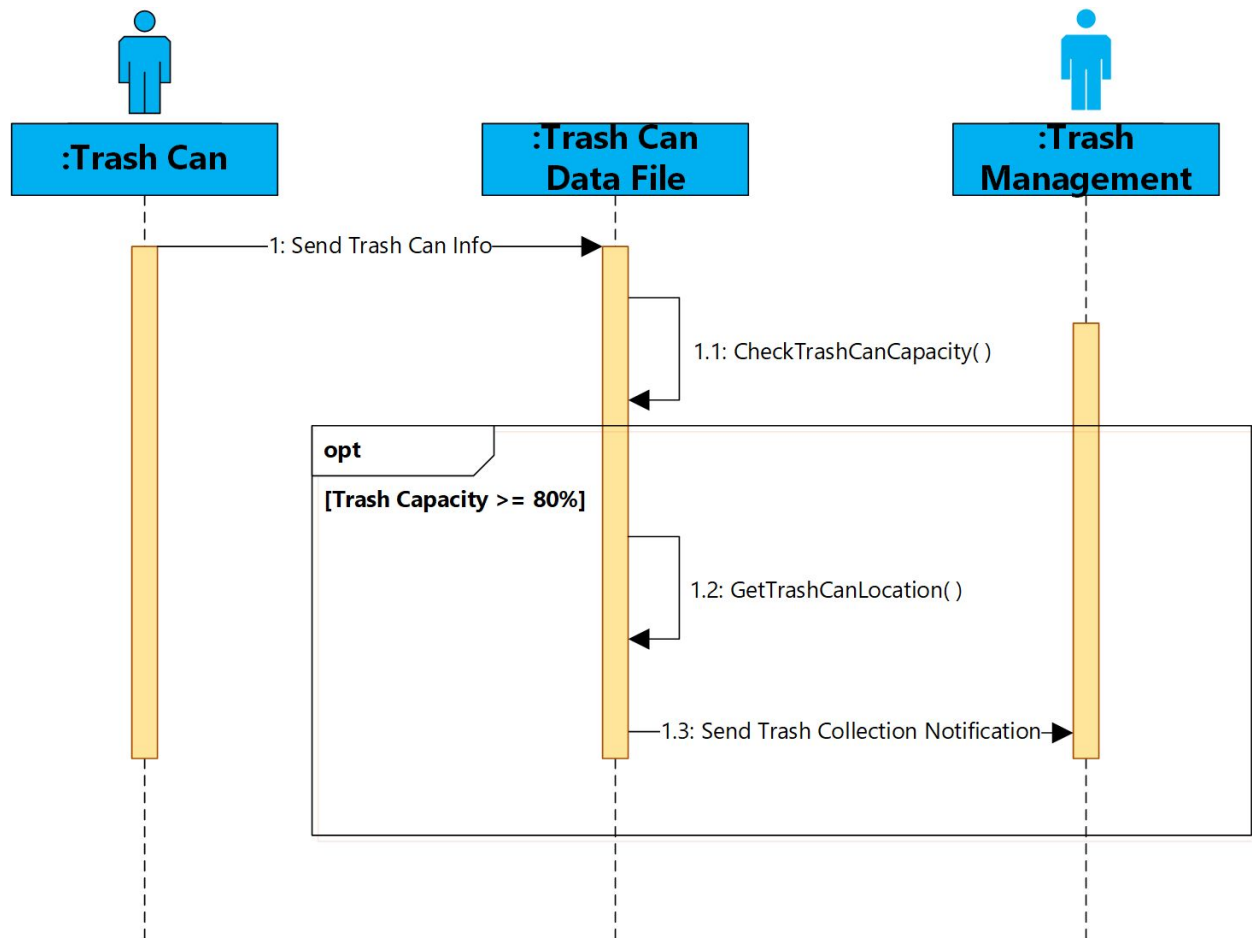


11. Sequence Diagram

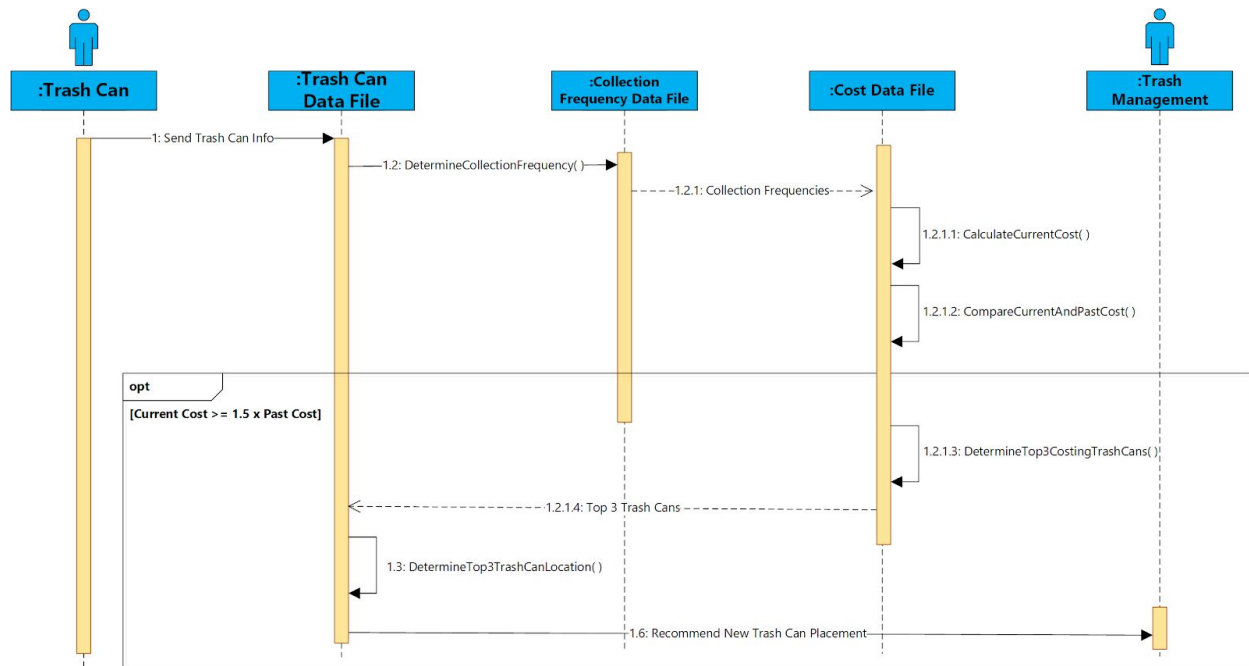
1. Find Trash Can Information:



2. Send Trash Can Notification



3. Recommend Trash Can Placement

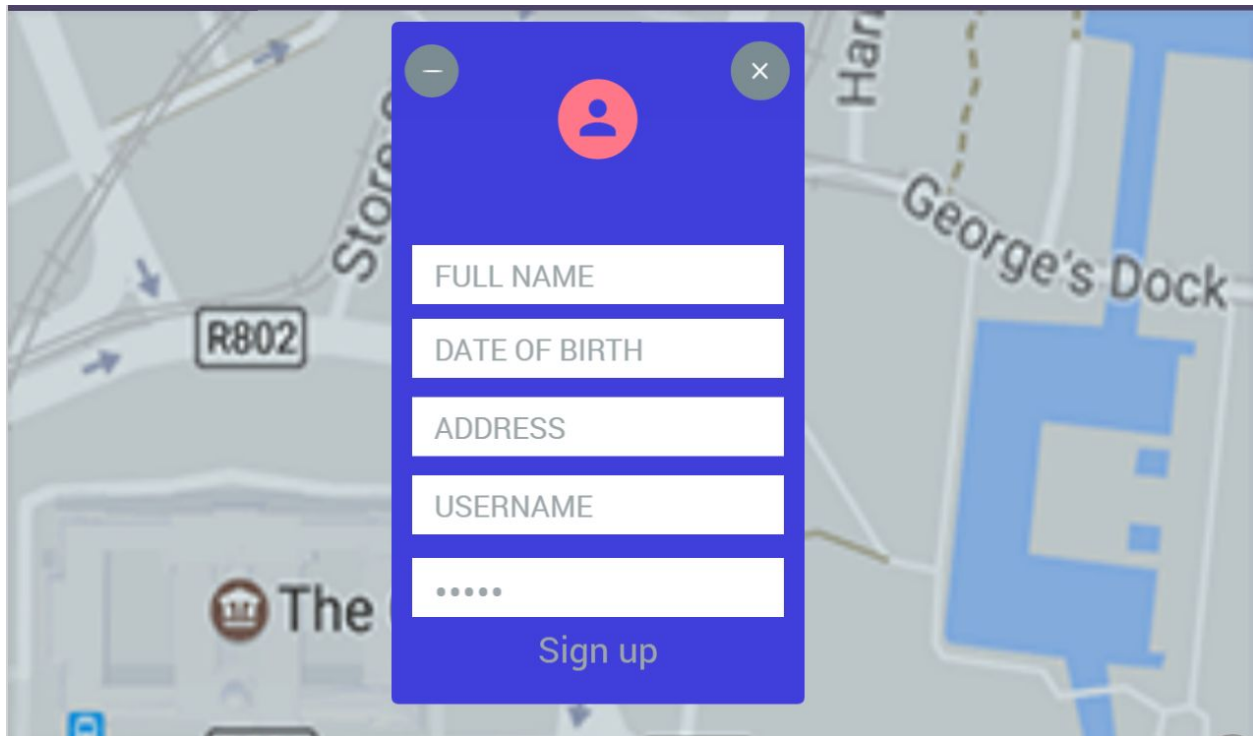


12. Functional Specification Document


- The proposed system will save a lot of time and money for trash management as the janitor just needs to collect the trash whenever he receives a notification for a trash can location. Therefore, the janitor does not need to periodically visit each trash can location and can instead just visit the locations that he was notified about.
- The proposed system will send a recommendation to trash management where there is a need of another trash can. The proposed system can cut down the cost of sending janitors too frequently to trash can locations that send many notifications. The system will analyze the trash collection frequency in order to calculate the current 6 months cost before comparing that cost with the past 6 months cost. Then, the system provides a recommendation for placement of a new trash can near the top 3 costing trash cans on this basis.
- The proposed system receives real time information about the trash can fill capacity of every trash can using the ultrasonic sensors and this data is used to update the trash can data file. This allows users the ability to access the most updated trash can information when searching for a nearby trash can, and it allows trash management to be updated quicker on trash cans that reach a fill capacity of 80% or above.
- The proposed system will make it easier for users to look for nearby trash cans and identify which trash can is best for trash disposal based on trash can fill capacity displayed on the website. This will reduce the time and effort of the user.
- The proposed system allows members to report a problem if they witness something wrong with a trash can. That problem report will be sent to trash management so that they can resolve the problem as early as possible to avoid inconvenience.

13. [Interface Design](#)

SIGN UP PAGE:



The image shows a mobile application interface for a sign-up page. A blue modal form is centered over a blurred map background. The form has a red circular profile icon at the top center. Below the icon are five white input fields with labels: 'FULL NAME', 'DATE OF BIRTH', 'ADDRESS', 'USERNAME', and a password field indicated by six dots. At the bottom of the form is a 'Sign up' button. The background map shows a street labeled 'R802' and a location named 'George's Dock'.



FULL NAME

DATE OF BIRTH

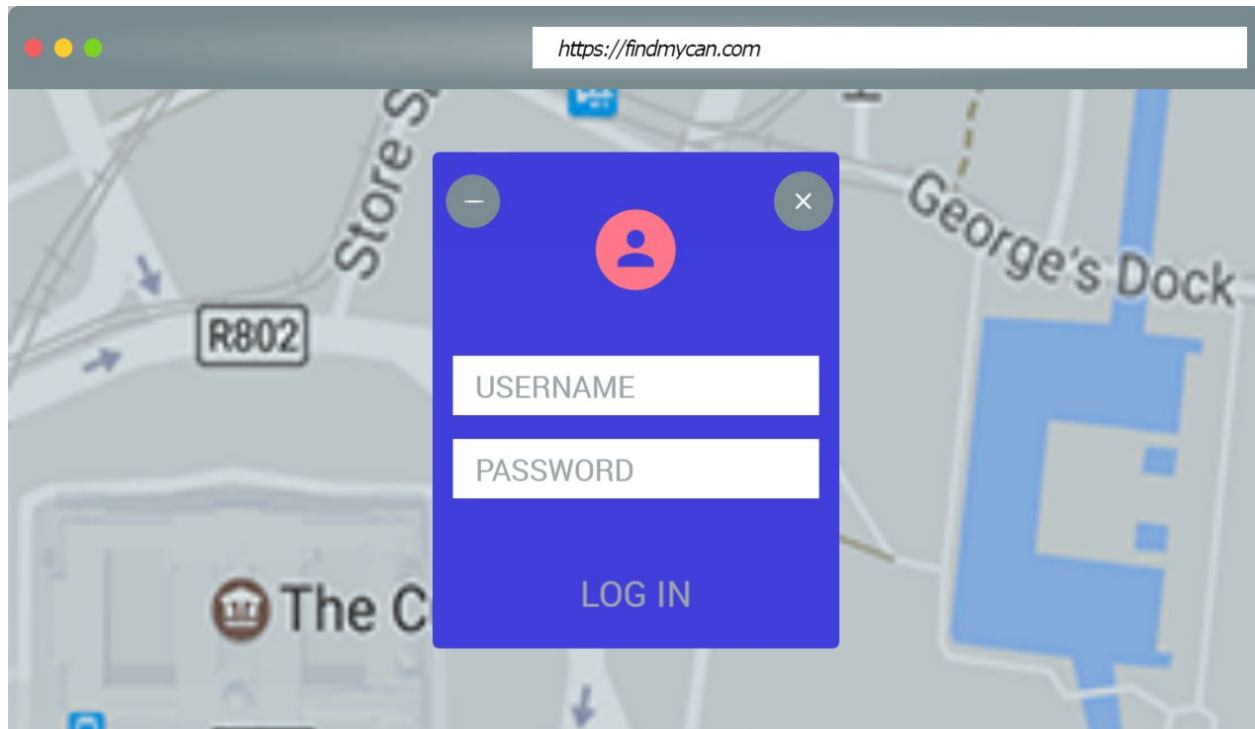
ADDRESS

USERNAME

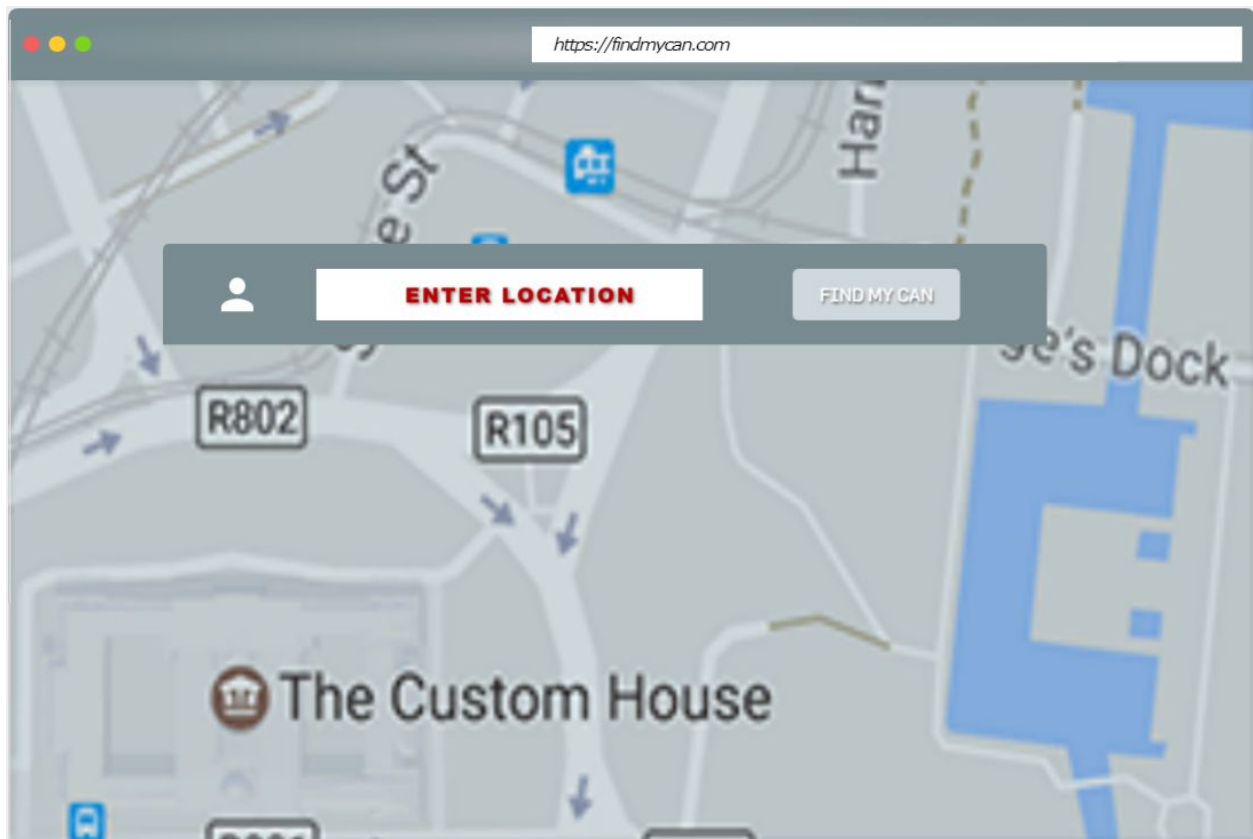
.....

Sign up

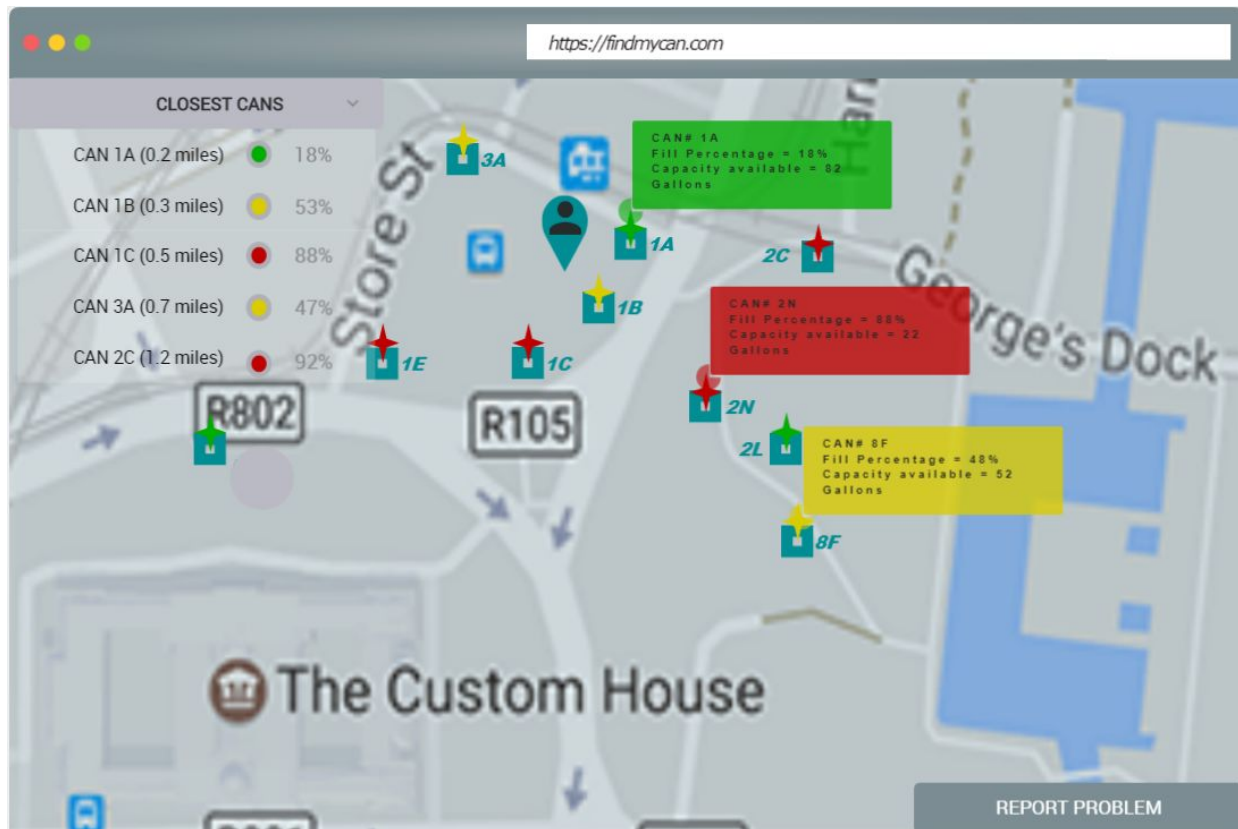
LOGIN PAGE:



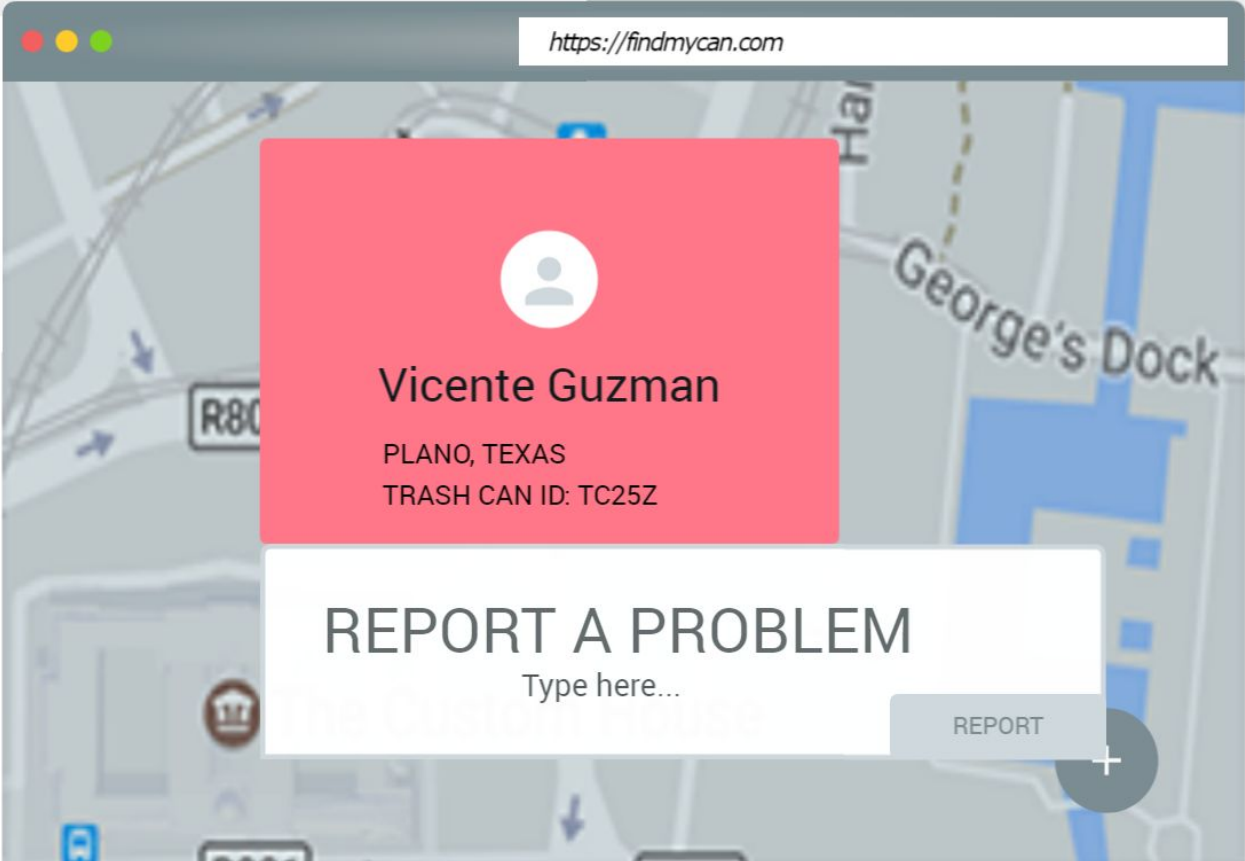
FIND TRASH CAN INFO:



REVIEW TRASH CAN INFO:




REPORT A PROBLEM:



The screenshot shows a web browser window with the URL `https://findmycan.com`. The background is a map of a city area, with labels like "George's Dock" and "R80" visible. Overlaid on the map is a red rectangular box containing a user profile. Below the profile is a white rectangular box with the text "REPORT A PROBLEM" and a text input field. To the right of the input field is a grey button labeled "REPORT".

<https://findmycan.com>



Vicente Guzman

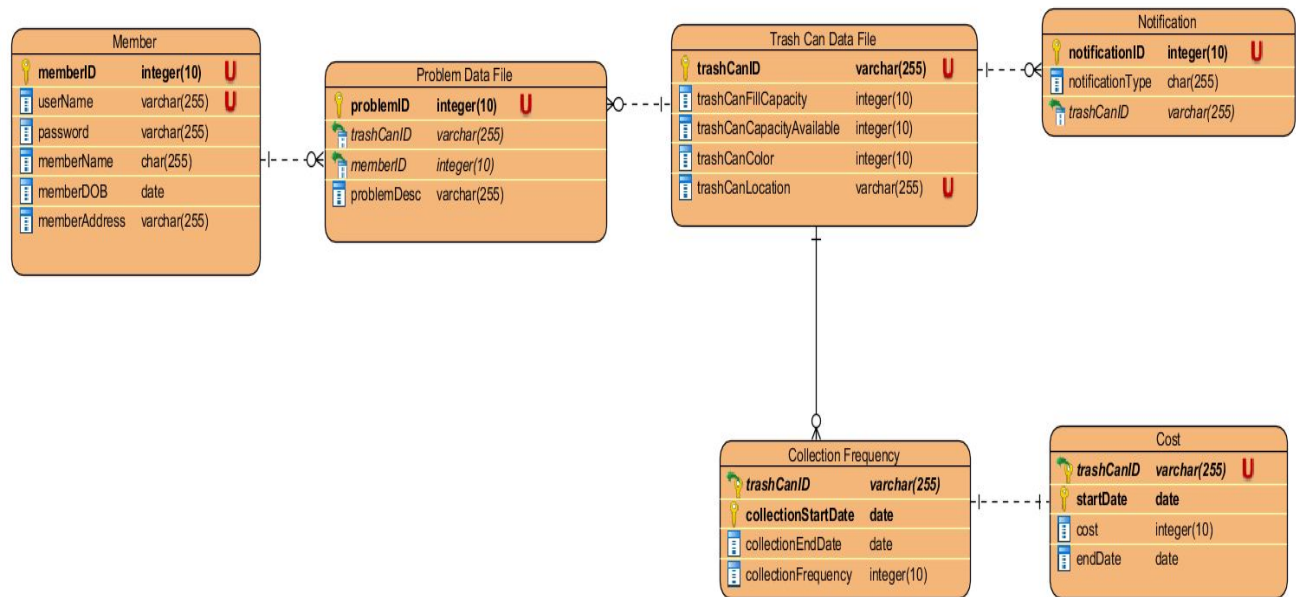
PLANO, TEXAS
TRASH CAN ID: TC25Z

REPORT A PROBLEM

Type here...

REPORT

14. Database Design:



Database Constraints

Member Table

- Primary key is “memberID”
- Primary key “memberID” and attribute “username” should be Unique
- Primary key “MemberID” and attributes “username”, “password”, “memberName”, “memberDOB”, “memberAddress” should not be Null

Trash Can Data File Table

- Primary key is “trashCanID”
- Primary key “trashCanID” and attribute “trashCanLocation” should be Unique
- Primary key “trashCanID” and attributes “trashCanFillCapacity”, “trashCanCapacityAvailable”, “trashCanColor” and “trashCanLocation” should not be Null

Problem Data File Table

- Primary key is “problemID”
- Foreign key “trashCanID” references primary key “trashCanID” of Trash Can Data File table
- Foreign key “memberID” references primary key “memberID” of Member table
- Primary key “problemID” should be Unique

- Primary key “problemID”, foreign key “trashCanID”, foreign key “memberID” and attribute “problemDesc” should not be Null

Notification Table

- Primary key is “notificationID”
- “trashCanID” is foreign key
- “trashCanID” references primary key “trashCanID” in Trash Can Data File table
- Primary key “notificationID” should not be Null

Collection Frequency Table

- “trashCanID” is primary key
- “trashCanID” is foreign key, references “trashCanID” from Trash Can Data File table
- “collectionStartDate” is primary key
- “trashCanID” and “collectionStartDate” is composite key
- “trashCanID”, “collectionStartDate”, “collectionEndDate” and “collectionFrequency” should not be Null

Cost Table

- “trashCanID” is primary key
- “trashCanID” is foreign key, references “trashCanID” from Collection Frequency table
- “startDate” is primary key
- “trashCanID” and “startDate” is composite key
- “trashCanID” is Unique
- “trashCanID”, “cost”, “startDate” and “endDate” should not be Null

15. Software Design:

1) Signature:

Method Name: Determine Trash Collection Frequency()	Class Name: Collection Frequency	ID: Collection Frequency ID
Clients (Consumers): Trash Management		
Associated Use Cases: Compare Trash Can Data, Process Trash Capacity		
Description of Responsibilities: Calculates the total trash collection frequency of individual trash cans over the period of 6 months		
Arguments Received: Trash Can ID, Trash Can Location, Collection Frequency		
Type of Value Returned: Total trash collection frequency over the period of 6 months of individual trash cans		
Pre-Conditions:		

Post-Conditions: Sending the calculated trash collection frequency of individual trash can over the 6 month period

1)Logic:

SET Current Date Period = Date.DatePeriod

FOR Current Date Period

DO

FETCH Trash Can ID FROM Collection Frequency Datafile

SET Current Trash Can ID = .Trash Can ID

FOR Current Trash Can ID

DO

COMPUTE = (Total collection frequency for the set trash can id)

2) Signature:

Method Name: Calculate Current Cost()	Class Name: Cost	ID: Cost ID
Clients (Consumers): Trash Management		

Associated Use Cases: Process Trash Capacity, Compare Trash Can Data
Description of Responsibilities: Calculates the total cost incurred by trash management for individual trash can for the current 6 month period
Arguments Received: Trash Can ID, Trash Can Location, Collection Frequency
Type of Value Returned: Total cost incurred by individual trash can over the current 6 month period
Pre-Conditions: Trash collection frequency of individual trash cans over the current 6 month period is required
Post-Conditions: Sending the total current cost of individual trash cans

2) Logic:

SET Current Date Period = Date.DatePeriod

FOR Current Date Period

DO

FETCH Trash Can ID FROM Collection Frequency Datafile

SET Current Trash Can ID = .Trash Can ID

FOR Current Trash Can ID

DO

CALCULATE Total Frequency for Trash Can ID

COMPUTE Current Cost = (Total collection frequency for the set trash can id)* (Cost incurred by trash management for collecting trash one time)

3) Signature:

Method Name: Compare Current and Past cost()	Class Name: Cost	ID: Cost ID
Clients (Consumers): Trash Management		
Associated Use Cases: Compare Trash Can Data, Process Trash Capacity		
Description of Responsibilities: Compare the current 6 months cost for each trash can incurred by trash management with the past 6 months cost of each trash can		

Arguments Received: Trash Can ID, Trash Can Location, Collection Frequency
Type of Value Returned: Trash Can ID, Trash Can Location, and Total Current Cost
Pre-Conditions: Current and past cost data of individual trash cans are required
Post-Conditions: Sends the Trash Can ID and Trash Can Location of the trash cans that have incurred 50% more cost than the past cost

3) Logic:

SET Current Date Period = Date.DatePeriod

FOR Current Date Period

DO

FETCH Trash Can ID FROM Cost Datafile

SET Current Trash Can ID = Trash Can ID

FOR Current Trash Can ID

DO

FETCH Total Current Cost FROM Cost Data File

SET Past Date Period = Date.DatePeriod

FOR Past Date Period

DO

SET Past Trash Can ID = Current Trash Can ID

FOR Past Trash Can ID

DO

FETCH Total Past Cost FROM Cost Data File

IF Total Current Cost $\geq 1.5 \times$ Total Past Cost

THEN

FETCH Trash Can ID FROM Trash Can Data File

FETCH Trash Can Location FROM Trash Can Data File

Else

PASS

4) Signature:

Method Name: Determine Top3 Costing Trash Cans()	Class Name: Cost	ID: Cost ID
Clients (Consumers): Trash Management		
Associated Use Cases: Process Trash Capacity, Compare Trash Can Data		
Description of Responsibilities: System determines the trash cans which incurred a current cost that is 1.5 times more than the past cost and then determines the top 3 costing trash cans.		
Arguments Received: Trash Can ID, Trash Can Location, Collection Frequency		
Type of Value Returned: Trash Can ID and Trash Can Location of top 3 costing trash cans		
Pre-Conditions: Data for trash cans that incurred a current cost that is 50% more than the past cost is required		

Post-Conditions: Sends the Trash Can ID and Trash Can Location of top 3 trash cans that have incurred the most cost to trash management

4) Logic:

IF Total Current Cost $\geq 1.5 \times$ Total Past Cost

SET the “Trash Can ID” as Key in Dict Max trash can value

SET the “Current Cost” as value in Dict Max trash can value

FETCH top 3 cost From Dict Max trash can value

5) Signature:

Method Name: Check Trash Can Capacity()	Class Name: Trash Can Data File	ID: TrashCanDataFile ID
Clients (Consumers): User, Trash Management		
Associated Use Cases: Find Trash Can Information, Process Trash Capacity, Send Trash Collection Notification		

Description of Responsibilities: Check if Trash Can Fill Capacity $\geq 80\%$
Arguments Received: Trash Can ID, Trash Can Location, Trash Can Fill Capacity
Type of Value Returned: Trash Can ID and Trash Can Location of Trash Can having fill capacity ≥ 80
Pre-Conditions:
Post-Conditions: Returns the trash can fill capacity value

5) Logic:

DO (every 1 hour)

FETCH Trash Can Fill Capacity FROM Trash Can

SET the value of "Trash Can Fill Capacity" in Trash Can Data File

IF Trash Can Fill Capacity is not $\geq 80\%$

THEN

SET the Notification Trash Collection = FALSE

ELSE

SET the Notification Trash Collection = TRUE

16. Project Management Deliverables

Meeting Timeline:

<u>Dates</u>	<u>Tasks</u>
September 12 (1 - 3pm)	Brainstorm project ideas for professor approval and exchange contact information
September 19 (1 - 3 pm)	Finalize project idea and start working on project statement
September 26 (1 - 3 pm)	Finalize project statement and build a project completion timeline
October 3 (1 - 5 pm)	Discuss Use cases and Context diagram
October 10 (1 - 5 pm)	Finalize Use cases and Context Diagram. Discuss BPMN diagram
October 17 (1 - 3 pm)	Finalize BPMN diagram and discuss Data Dictionary
October 24 (1 - 4 pm)	Discuss class and sequence diagrams
October 31 (1 - 3 pm)	Finalize class diagram, sequence diagram, and Data Dictionary
November 7 (1 - 3 pm)	Discuss functional specification document and interface design
November 14 (1 - 3 pm)	Discuss database design, software designs, and complete class diagram
November 21 (1 - 3 pm)	Discuss project management deliverables and finalize functional specification document and interface design
November 25 (1 - 5 pm)	Finalize database design software design, and complete class diagram. Create presentation and executive summary
December 1 (1 - 5 pm)	Finalize presentation, executive summary, and rest of report

Minutes of Meetings:

Meeting #1:

Meeting Date and Time:	September 12 (1 - 3pm)
Attendees:	Anmol Vyas, Aparna Satheesh, Ganesa Parmar, Shalini Singh, Sumeet Singh
Discussion:	Exchanged contact information and ideal project meeting times
Work Assigned:	Everyone needs to come up with about 1 - 2 project ideas for the next meeting
Action Items for Next Meeting:	Finalize project idea after presenting to professor on September 16

Meeting #2:

Meeting Date and Time:	September 19 (1 - 3 pm)
Attendees:	Anmol Vyas, Aparna Satheesh, Ganesa Parmar, Shalini Singh, Sumeet Singh
Discussion:	Finalized the project idea and brainstormed on the objectives for the project statement
Work Assigned:	Everyone has to go through the project guidelines and build a project timeline
Action Items for Next Meeting:	Finalize the Project Statement

Meeting #3:

Meeting Date and Time:	September 26 (1 - 3 pm)
Attendees:	Anmol Vyas, Aparna Satheesh, Ganesa Parmar, Shalini Singh, Sumeet Singh
Discussion:	Finalize the project statement and planned out the rest of the project work
Work Assigned:	Each member should brainstorm possible use case and context diagrams for next meeting
Action Items for Next Meeting:	Discuss Use Cases and Context Diagram

Meeting #4:

Meeting Date and Time:	October 3 (1 - 5 pm)
Attendees:	Anmol Vyas, Aparna Satheesh, Ganesa Parmar, Shalini Singh, Sumeet Singh
Discussion:	Review use cases and context diagram made by each group member
Work Assigned:	Build Use Case diagram and Context diagram on software (Anmol, Ganesa). Write Use Case Descriptions (Aparna, Shalini, Sumeet)
Action Items for Next Meeting:	Review Diagrams and discuss BPMN Diagram

Meeting #5:

Meeting Date and Time:	October 10 (1 - 5 pm)
-------------------------------	-----------------------

Attendees:	Anmol Vyas, Aparna Satheesh, Ganesa Parmar, Shalini Singh, Sumeet Singh
Discussion:	Review and finalize the Use Case, Context Diagram, and Use Case Description made by each member. Discuss BPMN diagrams.
Work Assigned:	Choreography Diagram (Anmol, Ganesa), Data Dictionary (Aparna, Shalini, Sumeet)
Action Items for Next Meeting:	Review Choreography Diagram and Data Dictionary

Meeting #6:

Meeting Date and Time:	October 17 (1 - 3 pm)
Attendees:	Anmol Vyas, Aparna Satheesh, Ganesa Parmar, Shalini Singh, Sumeet Singh
Discussion:	Finalize choreography diagram and data dictionary. Discuss class and sequence diagram
Work Assigned:	Everyone should brainstorm for class and sequence diagram
Action Items for Next Meeting:	Finalize class and sequence diagram

Meeting #7:

Meeting Date and Time:	October 24 (1 - 4 pm)
Attendees:	Anmol Vyas, Aparna Satheesh, Ganesa Parmar, Shalini Singh, Sumeet Singh
Discussion:	Review Class and sequence diagram
Work Assigned:	Each member is assigned to come up with interface ideas and functional specification document

Action Items for Next Meeting:	Discuss possible interface design
---------------------------------------	-----------------------------------

Meeting #8:

Meeting Date and Time:	October 31 (1 - 3 pm)
Attendees:	Anmol Vyas, Aparna Satheesh, Ganesa Parmar, Shalini Singh, Sumeet Singh
Discussion:	Choose the most feasible interface design
Work Assigned:	Each member will make parts of the finalized interface design on the software and draft a copy of functional specific document
Action Items for Next Meeting:	Discuss the next phase of the project

Meeting #9:

Meeting Date and Time:	November 7 (1 - 5 pm)
Attendees:	Anmol Vyas, Aparna Satheesh, Ganesa Parmar, Shalini Singh, Sumeet Singh
Discussion:	Review Functional specific document and finalize the document
Work Assigned:	Each member should study about database design, software designs, and complete class diagram
Action Items for Next Meeting:	Finalize database design

Meeting #10

Meeting Date and Time:	November 14 (1 - 5 pm)
-------------------------------	------------------------

Attendees:	Anmol Vyas, Aparna Satheesh, Ganesa Parmar, Shalini Singh, Sumeet Singh
Discussion:	Discuss database design, software designs, and complete class diagram
Work Assigned:	Each member will work on the database design on the software
Action Items for Next Meeting:	Finalize Complete class and Software design

Meeting #11

Meeting Date and Time:	November 21 (1 - 5 pm)
Attendees:	Anmol Vyas, Aparna Satheesh, Ganesa Parmar, Shalini Singh, Sumeet Singh
Discussion:	Review previous work and all the diagrams
Work Assigned:	Each member will write a single signature and logic of the software design
Action Items for Next Meeting:	Finalize Database Design, Functional Specific Document and Project Report

Meeting #12

Meeting Date and Time:	November 25 (1 - 5 pm)
Attendees:	Anmol Vyas, Aparna Satheesh, Ganesa Parmar, Shalini Singh, Sumeet Singh
Discussion:	Review Functional Specific Document and Project Report
Work Assigned:	Each member will work on the project presentation
Action Items for Next Meeting:	Review Project Presentation and Report

Meeting #13:

Meeting Date and Time:	December 1 (1 - 5 pm)
Attendees:	Anmol Vyas, Aparna Satheesh, Ganesa Parmar, Shalini Singh, Sumeet Singh
Discussion:	Complete the project presentation
Work Assigned:	None
Action Items for Next Meeting:	None

17. References

- Fluid UI for UI interface design
- Visual Paradigm for Database design
- MS Visio for BPMN, Context diagram, and Sequence diagram design
- “MeriTalk State & Local – State & Local IT News.” *MeriTalk State & Local – State & Local IT News*, <https://meritalkslg.com/>. or trash can information and trash facts used in presentation
- *Cleanair.org*, <https://cleanair.org/>. for trash facts used in presentation
- Google Slides for project presentation
- Google Doc for project report